文章编号:1672-3961(2024)04-0021-14 DOI:10.6040/j.issn.1672-3961.0.2023.148

基于贝叶斯优化的强化学习广义不动点解逼近

陈兴国[1,2],吕咏洲[1],巩宇[1],陈耀雄[3]
(1.南京邮电大学大数据安全与智能处理重点实验室, 江苏 南京 210023; 2.南京大学计算机软件新技术国家重点试验室, 江苏 南京 210046; 3.淮阴工学院电子信息工程学院, 江苏 淮安 223003)

摘要:针对强化学习不动点的解更优这一问题,提出广义不动点解模型设计,该设计使用 n 步自举法的不动点解扩展和基于线性插值法的不动点解构造方法。将该设计应用于成熟的 CBMPI 算法框架上,提出基于广义不动点的 CBMPI(n,β)算法。针对如何表达并逼近最优解这一问题,提出基于贝叶斯优化的广义不动点解的参数优化和基于集成学习的更高质量的解。在经典的 10×10 规模的 Tetris 游戏环境中验证算法提出的有效性。试验结果证明了基于线性插值法的广义不动点构造能比 n 步传统不动点效果好,其效果与其超参数步长 n 和插值参数 β 有很大关联。在 100 局的 Tetris 游戏中,平均分达到 4 388.3,表明贝叶斯优化技术可以找到多组表现优异的策略。对表现优异的四组广义不动点的策略参数(贝叶斯优化技术的结果)进行策略集成和值函数集成,得到更高质量的解。平均分可以分别达到 4 526.29 和 4 579.74,试验结果表明基于广义不动点的策略集成和基于广义不动点的值函数集成的分数相较于广义不动点的分数有小幅度提高,证实了可以通过集成学习寻找更高质量的解。

关键词:强化学习;值函数近似估计;不动点;贝叶斯优化;俄罗斯方块
中图分类号:TP311 **文献标志码**:A
引用格式:陈兴国,吕咏洲,巩宇,等. 基于贝叶斯优化的强化学习广义不动点解逼近[J].山东大学学报(工学版),2024,54(4):21-34.
CHEN Xingguo, LÜ Yongzhou, GONG Yu, et al. Bayesian optimization-based generalized fixed point approximation[J]. Journal of Shandong University (Engineering Science), 2024, 54(4):21-34.

Bayesian optimization-based generalized fixed point approximation

CHEN Xingguo[1,2], LÜ Yongzhou[1], GONG Yu[1], CHEN Yaoxiong[3]
(1. Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, Nanjing University of Posts and Telecommunications, Nanjing 210023, Jiangsu, China; 2. National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210046, Jiangsu, China; 3. Faculty of Electronic Information Engineering, Huaiyin Institute of Technology, Huaian, 223003, Jiangsu, China)

Abstract: A generalized fixed-point solution model was proposed to address the question of what kind of reinforcement learning fixed-point solution was better. This design employed the extension of fixed-point solutions using n-step bootstrapping and constructed fixed-point solutions based on linear interpolation. This design was applied to the mature CBMPI algorithm framework, introducing the CBMPI(n,β) algorithm based on generalized fixed-points. Addressing the issue of expressing and approximating the optimal solution, optimization of parameters for generalized fixed-point solutions was proposed based on Bayesian optimization, and higher-quality solutions through ensemble learning were suggested. The effectiveness of the proposed algorithms was verified in the classical 10×10 Tetris game environment. Experimental results showed that the generalized fixed-point construction based on linear interpolation had outperformed the traditional n-step fixed-point method, and its performance was significantly associated with hyperparameters such as the step length n and interpolation parameter β. Over 100 games of Tetris, an average score of 4 388.3 was achieved, which indicated that Bayesian optimization techniques could identify multiple sets of outstanding strategies. By integrating strategies from four sets of outstanding generalized fixed-point parameters (results from Bayesian optimization techniques) and

收稿日期:2023-11-20
基金项目:国家自然科学基金资助项目(62276142,62206133,62202240,62192783);科技创新 2030——"新一代人工智能"重大项目资助项目(2018AAA0100905);江苏省产业前瞻与关键核心技术竞争资助项目(BE2021028);深圳市中央引导地方科技发展资金资助项目(2021Szvup056)
第一作者简介:陈兴国(1984—),男,江苏南通人,讲师,硕士生导师,博士,主要研究方向为机器学习、强化学习、智能博弈。
E-mail:chenxg@ njupt.edu.cn

integrating value functions, higher-quality solutions were obtained. Average scores reached 4 526.29 and 4 579.74 respectively, which demonstrated that policy ensemble based on generalized fixed-points and value function ensemble based on generalized fixed-points marginally improved scores compared to other generalized fixed-point policies. This confirmed the potential of ensemble learning to discover higher-quality solutions.

Keywords: reinforcement learning; value function approximation; fixed point; Bayesian optimization; Tetris

0 引言

在当代强化学习研究中,往往存在着维度灾难问题,具有较大或无限状态空间。在这种情况下,值函数不能精确计算,必须采用远小于状态、动作个数的参数对值函数进行近似估计[1-2]。

若目标值函数 $\boldsymbol{V}_\pi$ 是已知的,只需将 $\boldsymbol{V}_\pi$ 投影到基函数张成的空间上,即可得到的最优近似解。其等价于 $\min_\theta \| \hat{\boldsymbol{V}} - \boldsymbol{V}_\pi \|_{\boldsymbol{\rho}}$,其中每个状态的误差根据 $\boldsymbol{\rho}$ 分布进行加权。最优解为 $\hat{\boldsymbol{V}} = \boldsymbol{\Pi}_{\boldsymbol{\rho}} \boldsymbol{V}_\pi$,其中,$\boldsymbol{\Pi}_{\boldsymbol{\rho}} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{D}_{\boldsymbol{\rho}} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{D}_{\boldsymbol{\rho}}$,$\boldsymbol{\Pi}_{\boldsymbol{\rho}}$ 是投影矩阵,$\boldsymbol{D}_{\boldsymbol{\rho}}$ 是对角矩阵,$\boldsymbol{D}_{\boldsymbol{\rho}}(i,i) = \boldsymbol{\rho}(i)$[1]。

大多数情况目标值函数 $\boldsymbol{V}_\pi$ 难以得知。如何准确近似评估值函数是一个至关重要的问题。线性方法是一种常用的值函数近似方法[3]。不动点的视角为这一方法提供了一种切入点。目前存在 TD 不动点、BR 不动点、TD(λ)不动点、斜影不动点等等。

文献[4]提出时序差分算法 temporal difference learning, TD)。文献[5]采用最小二乘的方法,提出了最小二乘时序差分算法(least squares temporal difference learning, LSTD)。文献[5]采用递归最小二乘的方法,提出递归最小二乘时序差分算法(recursive least squares temporal difference learning, RLSTD)。文献[6]采用贪心选择方式更新误差最显著的特征项对应的参数,提出了增量最小二乘时序差分算法(incremental least squares temporal difference learning, iLSTD)。这些算法在同策略(on-policy)的设置中都被证明收敛到 TD 不动点。在异策略的设置中,文献[7]提出了拟合 Q 迭代算法(fitted q iteration, FQI),使用线性估算器并运用最小二乘算法时,证明了 FQI 算法收敛于 TD 不动点,不保证稳定收敛[2]。文献[8-9]提出了基于梯度的时序差分算法(gradient temporal difference, GTD)。文献[10-11]提出了基于梯度的时序差分算法 2(GTD2)和带梯度修正的线性时序差分算法(linearTDwith gradient correction, TDC)。这些算法也都被证明收敛到 TD 不动点。TD 不动点写成期望的形式为 $E[\delta\boldsymbol{\phi}] = 0$。异策略 TD 算法有发散的风险。

Bellman 残差方法(Bellman residual, BR)是另一类线性近似评估值函数的方法。文献[12]提出了基于 Bellman 误差的残差梯度算法(residual gradient, RG),文献[13]指出该残差法收敛到 BR 不动点。BR 不动点写成期望的形式为 $E[\delta(\boldsymbol{\phi} - \boldsymbol{\gamma}\boldsymbol{\phi}')] = 0$。无论是在同策略还是异策略的情况下,Bellman 残差法均有严谨的收敛性保证,通常收敛的解不是最优解。

文献[4]提出了 TD(λ)算法,文献[14]证明该算法以概率 1 收敛到 TD(λ)不动点,其期望形式为 $E[\delta\boldsymbol{e}] = 0$ 的解,其中,e 是资格迹。文献[15]提出了 iLSTD 的改进 iLSTD(λ)算法。文献[16]提出了异策略 GQ(λ)算法。这些算法也都被证明收敛到 TD(λ)不动点。

文献[17]提出了混合最小二乘算法(hybrid least-squarestemporal difference learning, HLSTD),文献[18]提出了一般化斜投影的异策略算法(residual temporal difference learning, RTD),二者都收敛到一般化斜投影 Bellman 不动点解,写成期望的形式为 $E[\delta(\boldsymbol{\phi} - w\gamma\boldsymbol{\phi}')] = 0$,其中,$w$ 是超参数,但由于选取权值 w 的方式是通过人为设定的[18],存在无法表达并逼近最优解的问题;只考虑到单步,存在着误差。

上述不动点的解都不是最优的[2]。什么样的强化学习不动点的解更优、如何表达并逼近最优解是强化学习发展至今不得不直面的两个主要问题,也是本研究拟解决的问题。

1 相关背景

1.1 强化学习基本概念

强化学习所能解决的问题需要满足马尔科夫属性,即强化学习问题可以使用马尔科夫决策过程

(markov decision processes, MDPs)来建模[3]。

马尔科夫决策过程由一个四元组$\langle \boldsymbol{S},\boldsymbol{A},\boldsymbol{P},\boldsymbol{R}\rangle$组成,其中$\boldsymbol{S}$表示状态空间,$\boldsymbol{A}$表示动作空间,$\boldsymbol{P}:\boldsymbol{S}\times\boldsymbol{A}\times\boldsymbol{S}\rightarrow[0,1]$是状态转移函数,$\boldsymbol{R}:\boldsymbol{S}\times\boldsymbol{A}\rightarrow\boldsymbol{R}$是奖赏函数。策略$\boldsymbol{\pi}$是一个映射:$\boldsymbol{S}\times\boldsymbol{A}\rightarrow[0,1]$,它表示在状态$s$处选择动作$a$的概率。给定一个 MDP,强化学习智能体的目标是找到一个最优策略元$\boldsymbol{\pi}^*$来最大化一个长期的奖赏总和$\boldsymbol{R}_{\pi}$,定义如下[19]:

$$\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} \boldsymbol{R}_{\boldsymbol{\pi}}, \tag{1}$$

长期的累计奖赏和定义如下:

$$\boldsymbol{R}_{\boldsymbol{\pi}} = E_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right], \tag{2}$$

式中,γ是折扣因子,r_t是在时间步t的奖赏,$E_{\boldsymbol{\pi}}[\cdot]$是策略$\boldsymbol{\pi}$下的期望。

在强化学习方法中,值函数是极其重要的基本概念和学习方法,用来描述智能体处在状态s的好坏或者在状态s采取动作a的优劣,评价标准则是通过未来所获得期望奖赏值。给定的稳定策略$\boldsymbol{\pi}$,定义状态值函数如下[19]:

$$\boldsymbol{V}^{\pi}(s) = E_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty}\gamma^t r_t \mid s_0 = s\right], \tag{3}$$

式中$E_{\boldsymbol{\pi}}[\cdot\mid s]$表示初始状态为$s$的期望。

Bellman 方程是求解 MDP 问题的核心,可以被看作是一个函数的不动点方程。它可以表示为当前状态和下一状态之间的递归关系[20]:

$$\boldsymbol{V}^{\pi}(s) = \sum_{a}\boldsymbol{\pi}(a\mid s)\sum_{s'}\boldsymbol{P}(s,a,s')[\boldsymbol{R}(s,a) + \gamma\boldsymbol{V}^{\pi}(s')], \tag{4}$$

式中,$\boldsymbol{\pi}(a|s)$表示状态s选择动作a的概率,$\boldsymbol{P}(s,a,s')$表示状态s下选择动作a转移到状态s'的概率,$\boldsymbol{R}(s,a)$表示状态s选择动作a获得的奖励。

根据巴拿赫不动点定理,Bellman 算子是有限空间上的压缩映射, Bellman 方程的解就是值函数的不动点,当价值函数达到某个状态时,它们不再发生变化。Bellman 方程用于迭代求解状态值函数,通过迭代逼近算法求解。其中,策略迭代就是一种基于 Bellman 方程的迭代逼近算法。在策略迭代算法中,通过 Bellman 方程的迭代逼近,得到一个值函数的不动点,即最优值函数[20]。

1.2　基于不动点视角的强化学习

如何准确近似评估值函数是一个至关重要的问题。线性方法是一种常用的值函数近似方法。不动点的视角为这一方法提供了一种切入点。

线性方法由一组特征向量和参数线性求和而成。定义如下[19]:

$$\boldsymbol{V}_{\boldsymbol{\theta}} = \boldsymbol{\Phi}\boldsymbol{\theta}, \tag{5}$$

式中,$\boldsymbol{\Phi}$是特征向量,$\boldsymbol{\theta}$是待学习的与$\boldsymbol{\Phi}$同维的权重向量。

利用 Bellman 方程的思想,采用迭代法进行求解$\boldsymbol{V}_{\boldsymbol{\theta}}$,即$\boldsymbol{V}_{\boldsymbol{\theta}}^{n+1} = \boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}^{n}$。奖赏函数$\boldsymbol{R}$由 MDP 环境定义的[1],不受特征$\boldsymbol{\Phi}$空间约束[13],导致$\boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}} = \boldsymbol{R} + \gamma\boldsymbol{P}\boldsymbol{V}_{\boldsymbol{\theta}}$不在特征空间中,带近似估计的算子$\boldsymbol{T}$不满足压缩映射,即$\boldsymbol{V}_{\boldsymbol{\theta}} \neq \boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}$。采用一个投影算子$\boldsymbol{F}$将$\boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}$投影回特征空间[19],如图 1 所示。

Bellman 算子作用于$\boldsymbol{\Phi}$平面上的价值函数,得到一个平面外的价值函数,接着被投影算子$\boldsymbol{F}$投影回来$\boldsymbol{\Phi}$平面上。如果投影算子$\boldsymbol{F}$合适的话,不断迭代地重复该过程,会在$\boldsymbol{\Phi}$平面上收敛于投影不动点解,即$\boldsymbol{V}_{\boldsymbol{\theta}}^{n+1} = \boldsymbol{F}\boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}^{n}$。整个迭代过程如图 2 所示。

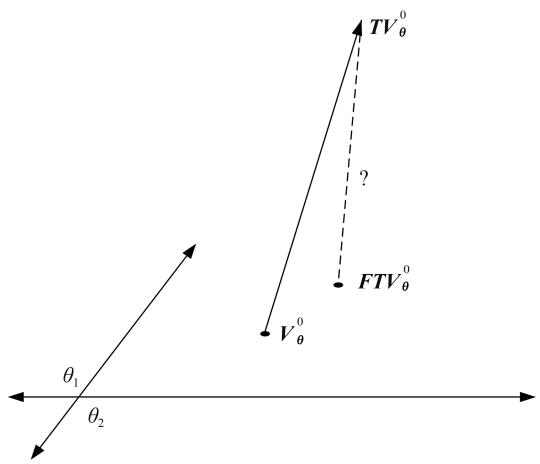

图 1　将$\boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}$投影到特征空间

Fig.1　Project the $\boldsymbol{T}\boldsymbol{V}_{\boldsymbol{\theta}}$ onto the feature space

从不动点视角出发,如何准确近似评估值函数问题的关键是构造合适的值函数投影不动点的解。

TD 不动点等价于[21]:$\min_{\boldsymbol{\theta}} \| \boldsymbol{V}_{\boldsymbol{\theta}} - \boldsymbol{\Pi T V}_{\boldsymbol{\theta}} \|_{\boldsymbol{D}}^2$,其中,$\boldsymbol{\Pi T V}_{\boldsymbol{\theta}}$ 是用正交投影 $\boldsymbol{\Pi}$ 将 $\boldsymbol{TV}_{\boldsymbol{\theta}}$ 投影回到 $\boldsymbol{\Phi}$ 平面上的映射点,$\boldsymbol{\Pi}=\boldsymbol{\Phi}(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{D}$。利用最小二乘的方法对其进行求解,得到 TD 不动点的解:

$$\boldsymbol{\theta}_{\mathrm{TD}}=(\boldsymbol{A}_{\mathrm{TD},\boldsymbol{D}_\rho})^{-1}\boldsymbol{b}_{\mathrm{TD},\boldsymbol{D}_\rho}=[\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{\Phi}-\gamma\boldsymbol{\Phi}')]^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{R},$$

$$\boldsymbol{A}_{\mathrm{TD},\boldsymbol{D}_\rho}=\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{D}_\rho(\boldsymbol{I}-\gamma\boldsymbol{P})\boldsymbol{\Phi},$$

$$\boldsymbol{b}_{\mathrm{TD},\boldsymbol{D}_\rho}=\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{D}_\rho\boldsymbol{R},$$

式中 $\boldsymbol{D}_\rho$ 表示状态分布。

BR 不动点[11]是另一类不动点。其通过最小化 Bellman 残差来进行求解,即,

$$\min_{\boldsymbol{\theta}} \| \boldsymbol{T}^{\pi}\hat{\boldsymbol{V}}-\hat{\boldsymbol{V}} \|,$$

作为 BR 不动点的典型代表,RG 算法通过梯度下降直接最小化加权 Bellman 误差,迭代为:

$$\boldsymbol{\theta}^{n+1}=(\boldsymbol{I}+\alpha\boldsymbol{A}_{\mathrm{BR},\boldsymbol{D}_\rho})\boldsymbol{\theta}^n+\alpha\boldsymbol{b}_{\mathrm{BR},\boldsymbol{D}_\rho},$$

式中,

$$\boldsymbol{A}_{\mathrm{BR},\boldsymbol{D}_\rho}=\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P})^{\mathrm{T}}\boldsymbol{D}_\rho(\boldsymbol{I}-\gamma\boldsymbol{P})\boldsymbol{\Phi},\ \boldsymbol{b}_{\mathrm{BR},\boldsymbol{D}_\rho}=\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P})^{\mathrm{T}}\boldsymbol{D}_\rho\boldsymbol{R}。$$

如果 $\boldsymbol{A}_{\mathrm{BR},\boldsymbol{D}_\rho}$ 可逆,则迭代式存在唯一的极限,则可以直接通过最小二乘方法的方式求解 $\boldsymbol{\theta}$。所得到的解为

$$\boldsymbol{\theta}_{\mathrm{BR},\boldsymbol{D}_\rho}=(\boldsymbol{A}_{\mathrm{BR},\boldsymbol{D}_\rho})^{-1}\boldsymbol{b}_{\mathrm{BR},\boldsymbol{D}_\rho}=(\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P})^{\mathrm{T}}\boldsymbol{D}_\rho(\boldsymbol{I}-\gamma\boldsymbol{P})\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P})^{\mathrm{T}}\boldsymbol{D}_\rho\boldsymbol{R}。$$

BR 不动点可以写成 $\boldsymbol{V}_{\boldsymbol{\theta}}=\boldsymbol{\Pi}_{\mathrm{BR}}\boldsymbol{T}^{\pi}\boldsymbol{V}_{\boldsymbol{\theta}}$,$\boldsymbol{\Pi}_{\mathrm{BR}}=\boldsymbol{\Phi}((\boldsymbol{\Phi}-\gamma\boldsymbol{\Phi}')^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}(\boldsymbol{\Phi}-\gamma\boldsymbol{\Phi}')^{\mathrm{T}}\boldsymbol{D}$。

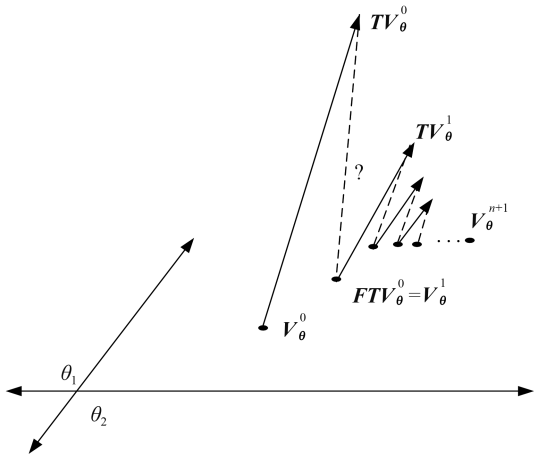

图 2 $\boldsymbol{V}_{\boldsymbol{\theta}}$ 收敛于不动点

Fig.2 $\boldsymbol{V}_{\boldsymbol{\theta}}$ converges at the fixed point

1.3 强化学习中的 n 步自举法

在基于蒙特卡洛的更新中,$\boldsymbol{v}_{\pi}(\boldsymbol{S}_t)$ 的估计值会沿着完整回报的方向进行更新,即

$$\boldsymbol{G}_t \doteq \boldsymbol{R}_{t+1}+\gamma\boldsymbol{R}_{t+2}+\gamma^2\boldsymbol{R}_{t+3}+\cdots+\gamma^{T-t-1}\boldsymbol{R}_T, \tag{6}$$

式中:T 是终止状态的时刻;$\boldsymbol{G}_t$ 是累积收益,也即回报。

不同于蒙特卡洛更新中的目标 G_t,在单步时序差分更新中的目标是即时收益加上后继状态的价值函数估计值乘以折扣系数,称之为单步回报,即

$$\boldsymbol{G}_{t:t+1} \doteq \boldsymbol{R}_{t+1}+\gamma\boldsymbol{V}_t(\boldsymbol{S}_{t+1}), \tag{7}$$

式中 $\boldsymbol{V}_t:\boldsymbol{S}\rightarrow\mathbf{R}$ 是在 t 时刻 $\boldsymbol{v}_{\pi}$ 的估计值。$\boldsymbol{G}_{t:t+1}$ 的下标表示这是一种截断回报,它由当前时刻 t 到时刻 $t+1$ 的累积收益和折后回报 $\gamma\boldsymbol{V}_t(\boldsymbol{S}_{t+1})$ 组成,其中 $\gamma\boldsymbol{V}_t(\boldsymbol{S}_{t+1})$ 替代了完整回报中的 $\gamma\boldsymbol{R}_{t+2}+\gamma^2\boldsymbol{R}_{t+3}+\cdots+\gamma^{T-t-1}\boldsymbol{R}_T$。

这种想法也能扩展到 n 步的情况。更新的目标是 n 步回报[19]:

$$\boldsymbol{G}_{t:t+n} \doteq \boldsymbol{R}_{t+1}+\gamma\boldsymbol{R}_{t+2}+\cdots+\gamma^{n-1}\boldsymbol{R}_{t+n}+\gamma^n\boldsymbol{V}_{t+n-1}(\boldsymbol{S}_{t+n}), \tag{8}$$

式中,$n\geqslant 1$,$0\leqslant t<T-n$。

在最坏的情况下,采用 n 步回报的期望值作为对 $\boldsymbol{v}_{\pi}$ 的估计可以保证比 $\boldsymbol{V}_{t+n-1}$ 更好。根据这个误差减少性质,可以证明所有的 n 步时序差分方法在合适的条件下都能收敛到正确的预测。n 步时序差分方法也存在着另一个优秀的性质,它是对单步时序差分法的拓展和衍生,单步时序差分方法和蒙特卡洛方法是 n 步时序差分方法两个极端特例。

1.4 CBMPI 算法

2015 年,Scherrer 猜想,好的策略比相应的价值函数能让智能体表现得更好,也更容易学习。因此,Scherrer 认为应该在策略空间中搜索 ADP 算法,而不是在值函数空间中搜索更传统的 ADP 算法。Scherrer 提出了基于分类的改进策略迭代算法(classification-based modified policy iteration, CBMPI)[22]。

CBMPI 算法是一种计算马尔可夫决策过程最优策略和最优值函数的迭代算法。算法的核心是回归器

和分类器。

回归器是指对值函数使用线性函数逼近,即 $\hat{\boldsymbol{v}}_k(s^{(i)})=\boldsymbol{\phi}(s^{(i)})\boldsymbol{\theta}$,其中 $\boldsymbol{\phi}(\cdot)$ 和 $\boldsymbol{\theta}$ 为特征向量和权重向量,将经验误差 $\hat{\mathscr{L}}_k^F(\hat{\mu};\boldsymbol{v})=\frac{1}{N}\sum_{i=0}^{N}(\hat{\boldsymbol{v}}_k(s^{(i)})-\boldsymbol{v}(s^{(i)}))^2$ 作为最小化目标,采用标准最小二乘法获得表现良好的 $\boldsymbol{\theta}$。

分类器训练集大小为 N,输入为 $s^{(i)}\in\boldsymbol{D}_k'$,输出 $(\max_a\hat{\boldsymbol{Q}}_k(s^{(i)},a)-\hat{\boldsymbol{Q}}_k(s^{(i)},a_1),\cdots,\max_a\hat{\boldsymbol{Q}}_k(s^{(i)},a)-\hat{\boldsymbol{Q}}_k(s^{(i)},a_{|\mathscr{A}|}))$。CBMPI 算法使用 $\boldsymbol{\pi}_w(s)=\operatorname{argmax}_a\boldsymbol{\psi}(s,a)^{\mathrm{T}}\boldsymbol{w}$ 这种形式的策略,其中 $\boldsymbol{\psi}$ 是策略特征向量(可能不同于值函数的特征向量 $\boldsymbol{\phi}$),$\boldsymbol{w}\in\mathscr{B}$ 是策略参数向量。通过最小化经验误差 $\hat{\mathscr{L}}_k^{\Pi}(\hat{\mu};\boldsymbol{\pi}_w)$,采用协方差矩阵自适应进化策略(CMA-ES)算法。

1.5　最优解逼近技术

贝叶斯优化是最优解逼近技术,是一种十分有效的全局优化算法,目标是找到全局最优解。贝叶斯优化方法在设计类问题上广泛应用。通过设计恰当的概率代理模型和采集函数,贝叶斯优化框架只需经过少数次目标函数评估即可获得理想解,非常适用于求解目标函数表达式未知、非凸、多峰和评估代价高昂的复杂优化问题[23]。

定理 1　贝叶斯定理 $\boldsymbol{p}(\boldsymbol{f}|\boldsymbol{D}_{1:t})=\frac{\boldsymbol{p}(\boldsymbol{D}_{1:t}|\boldsymbol{f})\boldsymbol{p}(\boldsymbol{f})}{\boldsymbol{p}(\boldsymbol{D}_{1:t})}$,式中:$\boldsymbol{f}$ 为黑箱函数;$\boldsymbol{D}_{1:t}$ 为已观测点的集合用,$\boldsymbol{D}_{1:t}=\{(\boldsymbol{x}_1,y_1)(\boldsymbol{x}_2,y_2)\cdots(\boldsymbol{x}_t,y_t)\}$;$y_t$ 为观测值,$y_t=\boldsymbol{f}(\boldsymbol{x}_t)+\varepsilon_t$,其中 $\boldsymbol{x}_t$ 为决策向量,ε_t 为观测误差;y 的似然估计为 $\boldsymbol{p}(\boldsymbol{D}_{1:t}|\boldsymbol{f})$,在观测的过程中必然会存在着无法避免的误差,将此误差命名为“噪声”;$\boldsymbol{p}(\boldsymbol{f})$ 为对黑箱函数状态的假设;$\boldsymbol{p}(\boldsymbol{D}_{1:t})$ 为边际化 $\boldsymbol{f}$ 的边际似然分布或者“证据”;$\boldsymbol{p}(\boldsymbol{f}|\boldsymbol{D}_{1:t})$ 表示 $\boldsymbol{f}$ 的后验概率分布。贝叶斯优化框架主要包含概率代理模型和采集函数两个核心部分。

集成学习是一种机器学习技术[24],基本思想是通过组合多个不同的学习算法来构建一个更加强大和稳健的分类器或回归器,提高预测准确性和鲁棒性,用于解决各种机器学习问题,例如分类、回归、聚类等。集成学习中学习算法可以是同质的(如同种类型的决策树)或异质的(如不同类型的分类器)。最常用的集成学习方法包括 Bagging、Boosting 和 Stacking 等。

Bagging(bootstrap aggregating)是一种基于 Bootstrap 采样技术的集成学习方法。在 Bagging 中,从训练数据集中随机有放回地抽取 n 条样本,组成一个新的训练数据集。用这个新的训练数据集训练多个基础模型,将预测结果进行投票或平均得到最终的预测结果。Bagging 算法能提高模型准确率和稳定性,降低结果的方差避免过拟合发生;减少噪音,体现样本真实的分布情况。

2　广义不动点的构造与逼近

现存的不动点解都不是最优的,什么样的强化学习不动点解更优,如何表达并逼近最优解是强化学习发展至今不得不直面的两个问题。

为解决上述问题,提出广义不动点解的构造,使用贝叶斯优化技术进行搜索和优化以逼近最优解,使用集成学习寻找更高质量的解。

2.1　广义不动点解的构造

广义不动点解的模型设计包括两个方面:从准确性角度出发,提出基于 n 步自举法的不动点解扩展;从投影角度出发,提出了基于线性插值法的不动点解构造。

TD 不动点和 BR 不动点都是基于单步回报。单步方法只考虑当前状态下的最优动作,忽略了未来状态和行动对当前状态下的影响,这使得一定程度上会存在着误差问题,容易对未来的预测不准确和受到随机性的影响,对于随机环境中的强化学习任务,可能会产生偏差。基于上述原因,广义不动点的模型设计的第一步是考虑使用强化学习中的 n 步自举法对 TD 不动点和 BR 不动点进行优化,将 TD 不动点和 BR 不动点从单步扩展到多步。

从统一的投影角度来理解 TD 不动点和 BR 不动点。TD 不动点是用正交投影 $\boldsymbol{\Pi}_{\mathrm{TD}}$ 将 $\boldsymbol{TV}_{\boldsymbol{\theta}}$ 投影到 $\boldsymbol{\Phi}$ 空间,而 BR 不动点则是用投影 $\boldsymbol{\Pi}_{\mathrm{BR}}$ 将 $\boldsymbol{TV}_{\boldsymbol{\theta}}$ 投影到 $\boldsymbol{\Phi}$ 空间。$\boldsymbol{\Pi}_{\mathrm{TD}}$ 和 $\boldsymbol{\Pi}_{\mathrm{BR}}$ 都满足斜投影的通项公式,可以看作是特殊的斜投影。从投影角度出发,广义不动点的模型设计的第二步考虑用线性插值的方式对 TD 不动点和 BR 不动点进行加权求和,将投影的方向一般化,使新的方向更接近最优投影方向。除了这一动机,还考虑另一个原因。TD 不动点算法倾向于产生比 BR 不动点算法更好的策略。这一特性的获得是以牺牲算法的稳定性为代价的,即无法稳定收敛。BR 不动点算法可以保证收敛到 BR 不动点,不如 TD 不动点算法预测准确,这二者都不是最优解。

基于上述原因,提出了广义不动点解的构造。

广义不动点通过线性插值的方法结合了 TD 不动点、BR 不动点和 n 步方法,即

$$\boldsymbol{A}_G=\beta\boldsymbol{A}_{\mathrm{BR}}^n+(1-\beta)\boldsymbol{A}_{\mathrm{TD}}^n, \tag{9}$$

$$\boldsymbol{b}_G=\beta\boldsymbol{b}_{\mathrm{TD}}^n+(1-\beta)\boldsymbol{b}_{\mathrm{TD}}^n, \tag{10}$$

整理一下,得

$$\boldsymbol{A}_G=\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\beta\gamma\boldsymbol{P}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}(\boldsymbol{I}-\gamma\boldsymbol{P}^n)\boldsymbol{\Phi}, \tag{11}$$

$$\boldsymbol{b}_G=\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\beta\gamma\boldsymbol{P}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}\boldsymbol{R}^n, \tag{12}$$

式中 $\boldsymbol{R}^n=\boldsymbol{R}_1+\gamma\boldsymbol{R}_2+\cdots+\gamma^{n-1}\boldsymbol{R}_n$。

广义不动点的最小二乘解为

$$\boldsymbol{\theta}_G=(\boldsymbol{A}_G)^{-1}\boldsymbol{b}_G=(\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\beta\gamma\boldsymbol{P}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}(\boldsymbol{I}-\gamma\boldsymbol{P}^n)\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{I}-\beta\gamma\boldsymbol{P}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}\boldsymbol{R}^n。 \tag{13}$$

写成期望的形式,广义不动点的解与 $E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}]=0$ 等价,其中,$\delta=r_1+\gamma r_2+\cdots+\gamma^{n-1}r_n+\gamma^n\boldsymbol{v}(s_n)-\boldsymbol{v}(s)$。通过定义可知,当 $\beta=1$ 的时候,广义不动点的解就退化成 BR 不动点的解,而当 $\beta=0$ 的时候,广义不动点的解就退化成 TD 不动点的解。在独立同分布问题中并且满足一些基本假设条件下,广义不动点的解以 1 的概率收敛到一般斜投影的 Bellman 不动点解[18]。

接下来证明这一等式。$\boldsymbol{V}_{\boldsymbol{\theta}}=\boldsymbol{\Phi}\boldsymbol{\theta}$,将 $E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}]=0$ 写成矩阵的形式,可以得到

$$(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}(\boldsymbol{R}^n+\gamma\boldsymbol{\Phi}^n\boldsymbol{\theta}-\boldsymbol{\Phi}\boldsymbol{\theta})=0, \tag{14}$$

将含 $\boldsymbol{\theta}$ 的项进行合并,整理得:

$$(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}(\boldsymbol{\Phi}-\gamma\boldsymbol{\Phi}^n)\boldsymbol{\theta}=(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}\boldsymbol{R}^n,$$

$$\boldsymbol{\theta}=((\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}(\boldsymbol{\Phi}-\gamma\boldsymbol{\Phi}^n))^{-1}(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}_{\boldsymbol{\rho}}\boldsymbol{R}^n, \tag{15}$$

式(15)与广义不动点的最小二乘解形式一致。

说明 $E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}]=0$ 称之为不动点解的原因。根据定义,$E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}]=0$ 可以进行如下等价变换:

$$0=E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}]=E_{\boldsymbol{\pi}}[(r_1+r_2+\cdots+\gamma^{n-1}r_n+\gamma\boldsymbol{\theta}^{\mathrm{T}}E_{\boldsymbol{\pi}}[\boldsymbol{\phi}^n]-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi})(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^n)^{\mathrm{T}}], \tag{16}$$

该式等价于:

$$0=\sum_s d_s[(r_1+r_2+\cdots+\gamma^{n-1}r_n+\gamma\boldsymbol{v}_{\boldsymbol{\theta}}(s_n)-\boldsymbol{v}_{\boldsymbol{\theta}}(s))(\boldsymbol{\phi}(s)-\beta\gamma\boldsymbol{\phi}(\boldsymbol{s}_n))^{\mathrm{T}}], \tag{17}$$

式中 $r_1+r_2+\cdots+\gamma^{n-1}r_n+\gamma\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{s}_n)$ 等价于用 n 步算子 $\boldsymbol{T}^{\boldsymbol{\pi},n}$ 作用于 $\boldsymbol{V}_{\boldsymbol{\theta}}$。写成向量的形式:

$$0=(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}(\boldsymbol{T}^{\boldsymbol{\pi},n}\boldsymbol{V}_{\boldsymbol{\theta}}-\boldsymbol{V}_{\boldsymbol{\theta}}), \tag{18}$$

该式等价于

$$(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{T}^{\boldsymbol{\pi},n}\boldsymbol{V}_{\boldsymbol{\theta}}=(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{V}_{\boldsymbol{\theta}}=(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi}\boldsymbol{\theta}, \tag{19}$$

得到

$$\boldsymbol{\theta}=((\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{T}^{\boldsymbol{\pi},n}\boldsymbol{V}_{\boldsymbol{\theta}}, \tag{20}$$

对等式两边同时左边点乘 $\boldsymbol{\Phi}$,可得

$$\boldsymbol{V}_{\boldsymbol{\theta}}=\boldsymbol{\Phi}\boldsymbol{\theta}=\boldsymbol{\Phi}((\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{T}^{\boldsymbol{\pi},n}\boldsymbol{V}_{\boldsymbol{\theta}}, \tag{21}$$

式中 $\boldsymbol{\Phi}((\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}$ 与投影的一般形式 $\boldsymbol{\Pi}_X=\boldsymbol{\Phi}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{X}^{\mathrm{T}}$ 一致,将 $\boldsymbol{\Phi}((\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\Phi})^{-1}(\boldsymbol{\Phi}-\beta\gamma\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{D}$ 记作 $\boldsymbol{\Pi}_G$,故,

$$\boldsymbol{V}_{\boldsymbol{\theta}}=\boldsymbol{\Phi}\boldsymbol{\theta}=\boldsymbol{\Pi}_G\boldsymbol{T}^{\boldsymbol{\pi},n}\boldsymbol{V}_{\boldsymbol{\theta}}, \tag{22}$$

$E_{\boldsymbol{\pi}}[\delta(\boldsymbol{\phi}-\beta\gamma\boldsymbol{\phi}^{n})^{\mathrm{T}}]=0$ 满足不动点解的定义,可以称其为广义不动点解。

2.2　基于广义不动点的 CBMPI(n,β)算法

为了更好将广义不动点应用于实践,将广义不动点解与 CBMPI 算法进行结合,提出了基于广义不动点的 CBMPI(n,β)算法。

CBMPI 算法在训练回归器时,从一个任意的初始策略$\boldsymbol{\pi}_1\in\boldsymbol{\Pi}$ 和$\boldsymbol{v}_0\in\boldsymbol{F}$ 开始。在每次迭代 k 中,建立一个新的 $\boldsymbol{v}_k$,作为 n 步 Bellman 算子$(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{v}_{k-1}$在 $\boldsymbol{F}$ 中的最佳逼近。为得到 $\boldsymbol{v}_k$,建立目标函数为$(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{v}_{k-1}$的回归问题。

通过对 N 个状态进行独立同分布抽样,从分布 μ^2 建立一个轨迹集 $\boldsymbol{D}_k$。对于任意的$s^{(i)}\in\boldsymbol{D}_k$,生成一条长为 n 的轨迹集$(s^{(i)},a_0^{(i)},r_0^{(i)},s_1^{(i)},\cdots,a_{n-1}^{(i)},r_{n-1}^{(i)},s_n^{(i)})$,其中 $a_n^{(i)}=\boldsymbol{\pi}_k(s_n^{(i)})$,且 $r_n^{(i)}$ 和 $s_{n+1}^{(i)}$是由这种行为选择引起的奖赏和下一个状态。计算出$[(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{v}_{k-1}](s^{(i)})$的无偏估计 $\hat{\boldsymbol{v}}_k(s^{(i)})$,形式如下:

$$\hat{\boldsymbol{v}}_k(s^{(i)})=\sum_{t=0}^{n-1}\gamma^{t}r_t^{(i)}+\gamma^{n}\boldsymbol{v}_{k-1}s_n^{(i)},\tag{23}$$

使用这个无偏估计来建立一个训练集$\{(s^{(i)},\hat{\boldsymbol{v}}_k(s^{(i)}))\}_{i=1}^{n}$。回归器使用该训练集计算$(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{v}_{k-1}$的估计值。即

$$\boldsymbol{\theta}=\operatorname{argmin}\frac{1}{N}\sum_{i=0}^{N}(\hat{\boldsymbol{v}}_k(s^{(i)})-\boldsymbol{v}(s^{(i)}))^{2}。\tag{24}$$

CBMPI 算法通过线性近似方法对值函数进行估计,使用回归器逼近真实值函数,形式上等价于 FQI 算法。在 FQI 算法中,使用线性估算器并运用最小二乘算法时,$\boldsymbol{\theta}$ 收敛于 TD 不动点。CBMPI 算法回归器逼近的值函数收敛于 TD 不动点。

将 TD 不动点替换成广义不动点。本研究提出了基于广义不动点的 CBMPI(n,β)算法,如算法 1 所示。

算法 1　基于广义不动点的 CBMPI(n,β)算法

输入　插值参数β,值函数空间 $\boldsymbol{F}$,策略空间 $\boldsymbol{H}$,状态分布 μ;

初始化　$\boldsymbol{\pi}_1\in\boldsymbol{\Pi}$ 是任意一种策略,$\boldsymbol{v}_0\in\boldsymbol{F}$ 是任意值函数;

for $k=1,2,\cdots,$do

构建一个轨迹样本集合(rollout set)$\boldsymbol{D}_k=\{s^{(i)}\}_{i=1}^{N}$

for all $s^{(i)}\in\boldsymbol{D}_k$ do

演绎新的轨迹并且返回 $\hat{\boldsymbol{V}}_k(s^{(i)})$

end for

构建一个轨迹样本集合(rollout set) $\boldsymbol{D}'_k=\{s^{(i)}\}_{i=1}^{N'}$

for all $s^{(i)}\in\boldsymbol{D}'_k$和动作 $a\in\boldsymbol{A}$ do

for $j=1,2,\cdots,M$ do

演绎新的轨迹并且返回 $\boldsymbol{R}_k^{j}(s^{(i)},a)$

end for

$$\hat{\boldsymbol{Q}}_k(s^{(i)},a)=\frac{1}{M}\sum_{j=1}^{M}\boldsymbol{R}_k^{j}(s^{(i)},a)$$

end for

用回归的方法近似值函数$\boldsymbol{V}_k\in\operatorname{argmin}\hat{\boldsymbol{L}}_k^{\mathscr{G}}=\|\boldsymbol{\Pi}_{X_\beta}(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{V}_{k-1}-\boldsymbol{V}\|_{2,\mu}^{2}$

用分类的方法近似贪心策略$\boldsymbol{\pi}_{k+1}\in\operatorname{argmin}\hat{\boldsymbol{L}}_k$

end for

从算法 1 中可以发现,在回归器训练中,值函数 $\boldsymbol{V}$ 的权重向量 $\boldsymbol{\theta}$ 收敛于广义不动点的解:

$$\boldsymbol{V}_k\in\operatorname{argmin}\hat{\boldsymbol{L}}_k^{\mathscr{G}}=\|\boldsymbol{\Pi}_{X_\beta}(\boldsymbol{T}_{\boldsymbol{\pi}_k})^{n}\boldsymbol{V}_{k-1}-\boldsymbol{V}\|_{2,\mu}^{2}。\tag{25}$$

对 N 个初始状态进行独立同分布抽样,从分布 μ^2 建立一个轨迹集 $\boldsymbol{D}_k$。对任意的$s^{(i)}\in\boldsymbol{D}_k$,进行特征化处理,得到 $\boldsymbol{\Phi}^{(i)}$。对任意的$s^{(i)}\in\boldsymbol{D}_k$生成一条长为 n 的轨迹集$(s^{(i)},a_0^{(i)},r_0^{(i)},s_1^{(i)},\cdots,s_t^{(i)},a_t^{(i)},r_t^{(i)},\cdots,a_{n-1}^{(i)},r_{n-1}^{(i)},s_n^{(i)})$,式中,$a_t^{(i)}=\boldsymbol{\pi}_k(s_t^{(i)})$,$\boldsymbol{\pi}_k$ 由策略参数 $\boldsymbol{w}_k$ 产生,$r_t^{(i)}$ 是由 $a_t^{(i)}$ 引起的奖赏。对任意一条轨迹, Tetris

游戏中,$\gamma=1$,得到 $r^{(i)}=r_0^{(i)}+r_1^{(i)}+\cdots+r_{n-1}^{(i)}$。将 N 条 $\boldsymbol{\Phi}^{(i)}$ 拼接,得到(N,x)的矩阵 $\boldsymbol{\Phi}=(\boldsymbol{\Phi}^{(1)}\ \boldsymbol{\Phi}^{(2)}\cdots\boldsymbol{\Phi}^{(N)})^{\mathrm{T}}$。其中 x 是特征的数量。将 N 条 $\boldsymbol{\Phi}^{(i,n)}$ 拼接,得到(N,x)的矩阵 $\boldsymbol{\Phi}^n=(\boldsymbol{\Phi}^{(1,n)}\ \boldsymbol{\Phi}^{(2,n)}\cdots\boldsymbol{\Phi}^{(N,n)})^{\mathrm{T}}$。将 N 条 $r^{(i)}$ 拼接,得到$(N,1)$的向量 $\boldsymbol{\Phi}^n=(r^{(1)}\ r^{(2)}\cdots r^{(N)})^{\mathrm{T}}$。

根据 2.1 节的推导,基于广义不动点解的值函数权重向量 $\boldsymbol{\theta}$ 的最小二乘形式为:

$$\boldsymbol{\theta}_G=((\boldsymbol{\Phi}^{\mathrm{T}}-\beta\boldsymbol{\Phi}^n)^{\mathrm{T}}(\boldsymbol{\Phi}^{\mathrm{T}}-\boldsymbol{\Phi}^n))^{-1}(\boldsymbol{\Phi}^{\mathrm{T}}-\beta\boldsymbol{\Phi}^n)^{\mathrm{T}}\boldsymbol{R}^n, \tag{26}$$

CBMPI(n,β)算法分类器训练和 CBMPI 算法类似,不再赘述。

2.3 逼近最优解

由 2.1 节可知,为了解决偏差问题和减少误差,综合 TD 不动点、BR 不动点、n 步方法各自的优点,构造了广义不动点的解以逼近真实值函数。

基于广义不动点解的一般表达式,需要优化参数(n,β)。不同的(n,β)对应的解各不相同,表现出效果也会不同。针对特定问题进行搜索和优化以逼近最优解,找到更好次优解,是接下来面临的重要问题。

网格搜索法会增加计算开销,效率不高。若想减小时间成本,需要增大步长,这又会使得算法效果无法得到保证,可能会有所“遗漏”。网格搜索法不适用于广义不动点解参数优化这一问题。

参数优化问题的第二种常见解决办法为梯度下降法。广义不动点参数优化问题经过建模,需要优化目标函数,无法写出具体的解析式。广义不动点嵌套于 CBMPI 算法中,给定输入(n,β),将产生一个环境反馈的最终得分作为输出,其它信息未知,这种优化问题被称为黑盒函数优化问题。常见的优化方法,例如牛顿法、拟牛顿法、共轭梯度法,都不适用于广义不动点解参数优化问题,这些方法都需要可知、可求导的目标函数。

进化算法是一类可以解决黑盒函数优化问题的参数优化算法。广义不动点解参数优化的优化目标函数具有以下特点:给定输入(n,β),不能立即产生一个输出,经过一定的等待时间才产生输出,这是由强化学习和 CBMPI 算法框架的特性决定的。从输入到输出非常耗时。在学术上,人们称具有这种性质的目标函数为昂贵函数。对于这种黑盒昂贵函数参数优化问题,研究人员无法知道最终的结果与待优化参数之间的直接联系。进化算法一般是基于种群的搜索方法,有时候为了得到目标函数的极值,需要进行上千次的评价推演。广义不动点解参数优化的优化目标函数是昂贵的,时间成本较大。进化算法效率低下,仍然不适用于解决该问题。

贝叶斯优化算法能解决黑盒昂贵函数优化问题,利用数据驱动优化的思想,在迭代优化过程中利用已评价过的解的信息,构建出一个代理模型,利用该代理模型逼近真实的目标函数。代理模型可以每次迭代过程中预测哪些解是潜在最优解,用真实的目标函数去评价这些潜在最优解,减少目标函数的评价次数,大大提高了效率。贝叶斯优化算法进行搜索和优化,寻找表现良好的(n,β)以逼近最优解。贝叶斯优化寻找(n,β)的算法如算法 2 所示。

算法 2 贝叶斯优化寻找 β 算法

输入 初始化点个数 n_0,最大迭代次数 N,代理模型 $\boldsymbol{g}(\beta)$,采集函数 $\alpha(x|\boldsymbol{D})$;

输出 最优候选评估点:$\{\beta^*,y^*\}$;

for $t=1,2,\cdots,$ do

最大化采集函数,得到下一个评估点:$\beta_t=\mathrm{argmax}_{\beta\in\chi}\alpha(\beta|\boldsymbol{D}_{1:t-1})$

$\boldsymbol{f}(\beta_t)=$ 基于广义不动点的 CBMPI(n_t,β_t)的测试得分函数

评估目标函数值 $y_t=\boldsymbol{f}(\beta_t)+\varepsilon_t$。

将新的观测数据$\{\beta_t,y_t\}$加入到 $\boldsymbol{D}_t$ 集合当中,更新概率代理模型。

end for

2.4 更高质量的解

不同的(n,β)对应的解各不相同,表现出的效果也不同。针对特定问题找到更高质量的解以逼近最优解,是接下来亟待解决的问题。

贝叶斯优化可以搜寻到表现优异的多组(n,β),每组(n_i,β_i)的模型对应一组策略参数 $\boldsymbol{\theta}_i$。考虑对表现最好的前 m 组 $\boldsymbol{\theta}_i$ 进行集成学习,找到更高质量的解。

用集成学习算法寻找更高质量的解分成两种方案,分别为策略集成和值函数集成。其中,策略集成是对不同 $\boldsymbol{\theta}_i$ 生成的不同策略 $\boldsymbol{\pi}_i$ 进行集成。值函数集成是对不同 $\boldsymbol{\theta}_i$生成的 $\boldsymbol{\theta}_i$表进行集成。策略集成算法如图

3 所示。

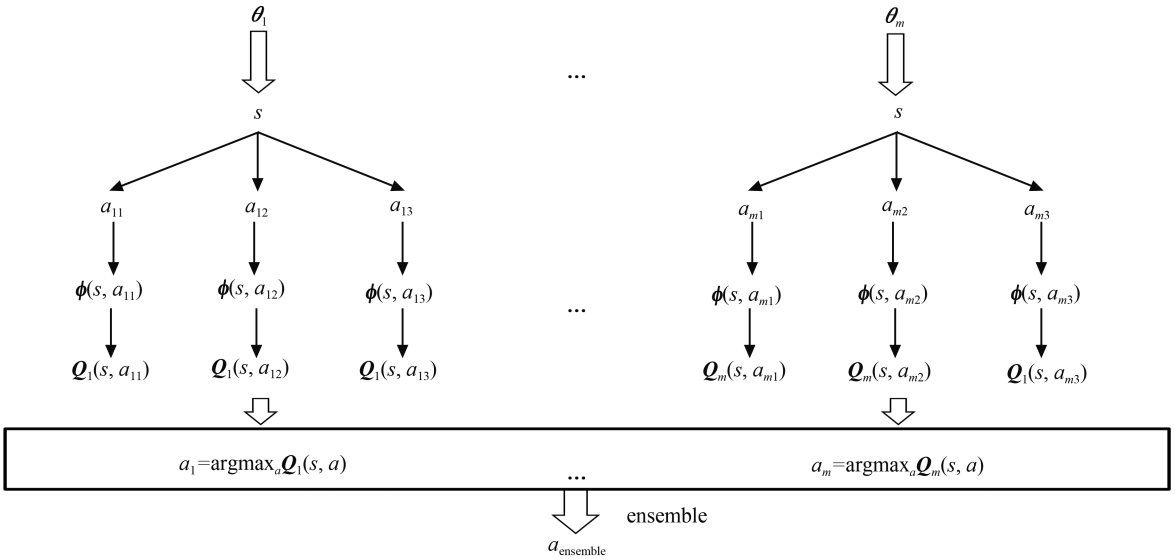

图 3　策略集成示意图

Fig.3　Policy integration diagram

每组 $\boldsymbol{\theta}_i$ 都能产生一张 $\boldsymbol{Q}_i$ 表,$\boldsymbol{Q}_i$ 表是一个二维数组,其中行表示智能体所处的状态,列表示智能体可以采取的行动。在每个状态 s 下,$\boldsymbol{Q}_i$ 表存储了每个行动的 $\boldsymbol{Q}_i(s,a_j)$,即智能体采取该行动可以获得的预期回报,$\boldsymbol{Q}_i(s,a_j)=\boldsymbol{\phi}(s,a_j)\boldsymbol{\theta}_{i_i}$,式中,$a_j$ 是状态 s 的一个可用动作,$\boldsymbol{\phi}(s,a_j)$是状态 s 和动作 a_j 的特征向量。根据 $\boldsymbol{Q}_i$ 表得到当状态为 s 的策略,即 $a_i=\text{argmax}_a\boldsymbol{Q}_i(s,a)$。$m$ 组 $\boldsymbol{\theta}_i$ 对应 m 组策略 a_i,对 m 组 a_i 做集成,集成方式基于投票决定,根据少数服从多数原则,选择出现次数最多的动作作为 a_{ensemble}。

算法 3　分类 Bagging 算法逼近最优解

输入　贝叶斯优化逼近最优解算法中表现最好的 m 个 β,状态 $s\in\boldsymbol{S}$;

输出　集成策略 a_{ensemble};

for $t=1,2,\cdots,$do

$\boldsymbol{Q}_i=$基于广义不动点的 CBMPI(β)的 $\boldsymbol{Q}$

根据 $\boldsymbol{Q}_i$ 表得到策略 $\boldsymbol{\pi}_i$,$a_i=\boldsymbol{\pi}_i(a|s)$

end for

对于 m 个回归模型 $C_1,C_2,\cdots,C_m$,每一个模型投票决定,少数服从多数原则,选择出现次数最多的动作作为 a_{ensemble}。值函数集成如图 4 所示。

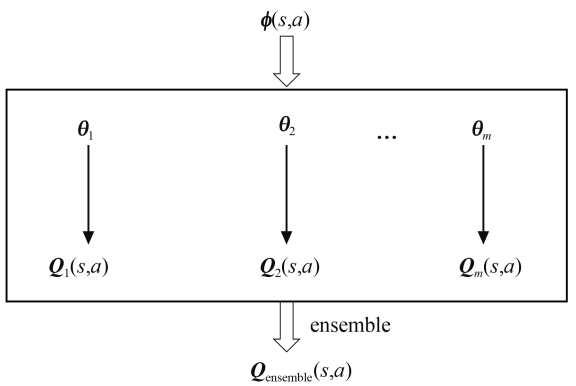

图 4　值函数集成示意图

Fig.4　Value function integration diagram

m 组 $\boldsymbol{\theta}_i$ 对应 m 张 $\boldsymbol{Q}_i(s,a)$ 表,对 m 张 $\boldsymbol{Q}_i(s,a)$ 表做集成,即将 m 张 $\boldsymbol{Q}_i(s,a)$ 表的同一位置做平均,得到 $\boldsymbol{Q}_{\text{ensemble}}(s,a)$ 表,根据 $\boldsymbol{Q}_{\text{ensemble}}(s,a)$ 表得到策略,即:$a=\text{argmax}_a\boldsymbol{Q}_{\text{ensemble}}(s,a)$。值函数集成算法如下所示:

算法 4 回归 Bagging 算法逼近最优解

输入 贝叶斯优化逼近最优解算法中表现最好的 m 个 β;状态 $s \in \boldsymbol{S}$;

输出 集成值函数 $\boldsymbol{Q}_{\text{ensemble}}$ 和集成策略 a_{ensemble};

for $t=1,2,\cdots,$do

$\boldsymbol{Q}_i$ = 基于广义不动点的 CBMPI(β)的 $\boldsymbol{Q}$

end for

$\boldsymbol{Q}_{\text{ensemble}} = \text{average}(\boldsymbol{Q}_1, \boldsymbol{Q}_2, \cdots, \boldsymbol{Q}_m)$

根据 $\boldsymbol{Q}_{\text{ensemble}}$ 表得到策略 $\boldsymbol{\pi}$, $a = \boldsymbol{\pi}_i(a \mid s)$

3 试验验证与结果分析

通过试验证明多步 TD 不动点的解和多步 BR 不动点的解不是值函数的最优近似解。证明提出的广义不动点的解比上述两种解更优。

10×10 的 Tetris 游戏相对于 10×20 的 Tetris 游戏具有更少状态空间,更快地训练出稳定的策略,更加紧凑,可以更容易地观察和调试每一个状态和行为,本研究采用 10×10 规模的 Tetris 游戏作为试验环境。Tetris 是一个测试不同强化学习算法的理想环境。Tetris 有一个明确的目标:消除方块以获得高分。这使得强化学习的目标明确,使得智能体在游戏中获得积极反馈。Tetris 游戏的状态动作空间较小,复杂性很高,使得智能体需要学习采取不同决策来达到最佳得分。

目前有多种算法应用于 10×10 的 Tetris 游戏环境,表现优异。典型代表为 1.4 节介绍的 CBMPI 算法,其回归器值函数收敛于 TD 不动点,能够得到 4 200 分。笔者在复现原论文的时候,无法得到如此高的分数,只能得到 3 484.76 分,故在后续对比试验中,采取 3 484.76 分作为 CBMPI 的实际结果,以作对比。文献[25]提出的 M-learning 算法,能达到 8 000 分。该算法具体分为值函数评估和策略改进两步。在值函数评估模块,采用多步截断回报作为近似目标,在策略改进模块,采用多分类回归的方法得到优秀的策略。笔者在阅读该论文源代码时,发现该论文的试验环境并非传统意义上的 Tetris 游戏环境,而是新式 Tetris 游戏环境,即下落棋子并非传统意义上的随机产生,是有“组”的概念,每组方块由 7 个随机选择的方块组成,按照固定的顺序下落,每组之间的方块类型各不相同。当采用传统的 Tetris 游戏作为试验环境后,M-learning 算法只能得到 2 300 分左右。

3.1 Tetris 游戏介绍

Tetris 游戏[26]——“俄罗斯方块”游戏,是一款著名的电脑游戏,最初由俄罗斯数学家阿列克谢·帕基特诺夫(Alexey Pajitnov)编写,随后在几乎所有电脑和游戏平台上发布。棋盘大小通常有 10×10 和 10×20 两种规模,试验中,选取了 10×10 的规模。

在 Tetris 游戏中有 7 种棋子分别被标记为“O”、“J”、“L”、“Z”、“S”和“T”。游戏规则如下:玩家控制从棋盘顶部落到底部的棋子的方向和水平位置,通过消除完整的水平线的方式来得分,这些线被移出游戏,使位于更高位置的棋子向下移动。倘若玩家在放置一个棋子,棋子落下之后所在的高度已经超过了棋盘的高度,那么此时游戏结束。

为了完成 Tetris 游戏的试验框架,需要具体化第 1 章中提到的马尔可夫决策过程中的元素,方便后续工作的展开。通过对元组$\langle \boldsymbol{S}, \boldsymbol{\Phi}, \boldsymbol{V}, \boldsymbol{Q}, \boldsymbol{A}, \boldsymbol{\pi}, \boldsymbol{R}, \boldsymbol{P}, \gamma \rangle$的解释来简单说明这一过程。

状态 $\boldsymbol{S}$ 指棋盘局面。值得注意的是状态 $\boldsymbol{S}$ 包含了下落棋子的信息。由于每个方格只有两种状态 0 或者 1,二进制数据本身信息量不大,需要将 $\boldsymbol{S}$ 转换为更能表示环境状态的特征 $\boldsymbol{\Phi}$。

特征 $\boldsymbol{\Phi}$ 是对游戏中出现的与游戏得分密切相关的棋盘形状的统称,通过特征函数,可以对游戏棋盘的一些有意义的特征进行数值计算。常见特征集如下:

(1) Dellacherie- thiery (D-T) Features:该集合包含了 Dellacheri 的 6 个特征与 Thiery 在 2009 年提出的 3 个特征[27-28]。值得注意的是,这是最常见的俄罗斯方块的特征集,具体见表 1。

表 1　D-T 的特征
Table 1　Characteristics of D-T

特征	描述
Landing height	一个方块被消除后,棋盘重心距离底部的距离
ErodedPieceCells	下落一个方块之前,每一列中已经有多少个方块被填充
RowTransitions	相邻行的状态转换数(从空到满或从满到空)
Columntransitions	相邻列的状态转换数(从空到满或从满到空)
Holes	孔的数量(至少被一个满格盖住的空白方格)
BoardWells	井的深度连加和(左右两边均不空的列为井)
HoleDepth	所有孔的高度之和
RowsWithHoles	含有孔的列数之和
Diversity	各列形状的差异值

(2) RBF 高度特征:这 5 个新特征定义为:$\exp\left[\left(-|c-ih/4|^2\right)/\left(2(h/5)^2\right)\right]$,其中 c 是列的平均高度,$h=10$,是棋盘的总行数。

Tetris 游戏状态数量巨大,值函数近似是最佳选择。Tetris 游戏的值函数定义为:

$$\boldsymbol{V}_{\boldsymbol{\theta}}(s)=\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(s)=\sum_i \boldsymbol{\phi}_i(s)\boldsymbol{\theta}_i,$$

式中 $\boldsymbol{\phi}_i(s)$ 表示状态 s 的特征,$\boldsymbol{\theta}$ 是特征权重。

动作状态价值函数 $\boldsymbol{Q}(s,a)$ 定义为:

$$\boldsymbol{Q}(s,a)=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\psi}(s,a)=\sum_i \boldsymbol{\psi}_i(s,a)\boldsymbol{w}_i,$$

动作价值函数的含义为在 s 局面下进行了动作 a 到达局面 s'这一事件的好坏。$\boldsymbol{\psi}(s,a)$ 描述的是 s 局面下进行了动作 a 到达局面 s'这一事件的特征,它与 $\boldsymbol{\phi}(s')$ 可以相同,也可以不相同。

$\boldsymbol{A}$ 指当前局面下,棋子出现时能进行的动作的集合,包括数次的旋转和平移。7 种棋子各自的动作集合不相同。

$\boldsymbol{P}(s,a,s')$ 表示进行了动作 a 后,状态 s 转换到下一个状态 s'的概率,在 Tetris 游戏中,这个概率显然恒为 1[28]。

$\gamma\in[0,1]$ 为折扣参数,游戏规则规定所有的分数均在消除满行后立即获得,未来的奖励值和立即奖励值等价,γ 恒为 1[28]。

$\boldsymbol{\pi}$ 表示智能体的策略,即 $\boldsymbol{\pi}:\boldsymbol{S}\rightarrow\boldsymbol{A}$,即智能体根据当前的环境状态 s 选择下一步的动作 a 的准则。在 Tetris 游戏中,策略通常采用贪心策略,定义为

$$\boldsymbol{\pi}(a|s)=\operatorname{argmax}\boldsymbol{Q}(s,a),\tag{27}$$

即在 s 局面下,每次都采取让 $\boldsymbol{Q}(s,a)$ 最大的 a 作为动作。

$\boldsymbol{R}(s,a)$ 是奖励函数,表示在状态 s 下进行动作 a 所能获得的奖励值。在 Tetris 游戏,这个函数的值等于一个棋子落下后消除的行数[28]。

MDP 框架中,状态由当前棋盘状态加上下落的棋子定义,动作定义为棋子的可行方向和它可以放置在棋盘上的可能位置,奖励的定义是最大化每个状态的期望奖励总和,与最大化该状态的分数相一致。Tetris 的状态空间很大,使用值函数线性近似方案,尝试从游戏训练中调整值函数参数(即权重)。

3.2　试验设置和结果分析

CBMPI (β)设定 β 为[0,1],贝叶斯搜索的迭代次数为 10。为保证试验的准确性,搜的目标函数为 100 局 Tetris 游戏结束时的平均消除行数。将 100 局 Tetris 游戏结束时的平均消除行数记为 σ,σ 越大,表示算法效果越好。

每次搜索迭代周期内迭代 10 次。每次迭代都会训练回归器和分类器,得到一组评估权重和策略权重。其中,演绎轨迹长度采用 CBMPI 中的最优参数 $n=5$,一共采集 $N=42\ 000$ 条样本数据,初始值函数参数设为 $\boldsymbol{\theta}=(0\ 0\ \cdots\ 0)^{\mathrm{T}}$,随机选取初始策略 $\boldsymbol{\pi}_1$,CMA-ES 参数(分类器参数)设置为 $\zeta=0.5$ 和 $\eta=0$。在回归器训练中,特征化方法为 D-T 特征加上 5 个 RBF 特征,在分类器训练中,特征化方法为 D-T 特征[22]。

通过大量试验,得到几组平均分数较的 β 值,具体如表 2 所示。

表 2 广义不动点中表现优异的几组 β 极其对应的平均分数

Table 2 Several groups of β with high performance in generalized fixed points and their corresponding average scores

方法	σ	方法	σ
广义不动点($n=5$, $\beta=0.41$)	4 006.81	广义不动点($n=2$, $\beta=0.53$)	ed3 319.265
广义不动点($n=7$, $\beta=0.21$)	4 138.49	广义不动点($n=4$, $\beta=0.24$)	2 458.4
广义不动点($n=8$, $\beta=0.93$)	4 154.41	广义不动点($n=6$, $\beta=0.12$)	2 477.875
广义不动点($n=5$, $\beta=0.72$)	4 388.33	广义不动点($n=8$, $\beta=0.95$)	3 233.59
广义不动点($n=5$, $\beta=0.36$)	3 154.14	广义不动点($n=3$, $\beta=0.67$)	3 346.01

由表 2 中可以看出,在 $n=5$,$\beta=0.72$ 时,其平均分数为 4 388.33,得分最高,在接下来的对比试验中,取 $n=5$,$\beta=0.72$。

$n=5$,$\beta=0.72$,广义不动点算法对应的回归器参数和策略参数如表 3 所示。

表 3 广义不动点分数最高的结果对应的回归参数和策略参数

Table 3 The results with the highest generalized fixed point score corresponds to the regression parameters and strategy parameters

特征	价值权重	策略权重	特征	价值权重	策略权重
Landing height	−0.027 6	−3.145 4	Rows/holes	−0.475 8	−10.350 2
Erode piece cells	0.347 6	1.425 9	Diversity	0.051 5	1.387 2
Row transitions	−0.046 0	−2.450 7	RBF1	0.822 4	
Column transitions	−0.048 5	−1.283 0	RBF2	2.856 6	
Holes	0.090 3	−2.805 9	RBF3	2.886 8	
Board wells	−0.013 6	−2.125 4	RBF4	3.718 4	
Hole depth	0.007 5	−1.256 3	RBF5	4.842 6	

注:参数 RBF1~5 五个特征只在计算价值函数中使用。

TD 不动点(CBMPI)、BR 不动点、广义不动点的最高平均分数如表 4 所示。由表 4 可知,本研究提出的广义不动点方法得分最高,能够比 CBMPI 算法的得分高出 903.57 分,取得了不错的进展。广义不动点 TD 不动点、BR 不动点、n 步方法各自的优点,能够解决偏差问题和减少误差。

表 4 TD 不动点、BR 不动点、广义不动点的最高平均分数对比

Table 4 Comparison of the highest mean scores and standard deviations of TD fixed point, BR fixed point, and generalized fixed point

方法	σ
TD 不动点(CBMPI)	3 484.76
BR 不动点	3 253.51
广义不动点($\beta=0.72$)	4 388.33

对上述表现良好的 4 组参数作策略集成和值函数集成,试验结果见表 5。

表 5 策略函数集成与值函数集成平均分数

Table 5 Strategy functions integrate mean scores and standard deviations with value functions

项目	参数 1	参数 2	参数 3	参数 4	策略集成	值函数集成
平均分数	4 006.81	4 437.16	4 138.49	4 154.41	4 526.29	4 579.74

通过试验结果,可以看到相较于任何一个参数的结果,经过策略集成和值函数集成后的平均分数都有小幅度提高,可见策略集成和值函数集成起到了作用,是行之有效的方法。

4 结论

线性值函数估计方法是一种基于函数逼近的值函数估计方法。在强化学习中,值函数不能被直接观察到,这使得估计问题非常困难。不动点的视角为如何准确近似评估线性值函数提供了一种切入点。通过定义一个映射函数来描述值函数的迭代过程,将值函数的估计转化为寻找该映射函数的不动点的问题。通过迭代计算,可以逐步逼近该不动点,得到更准确的值函数估计结果。现有的强化学习不动点的解均不是最优解且均存在各自的缺陷与不足。

根据以上分析,针对什么样的强化学习不动点的解更优这一问题提出了广义不动点解的模型设计以逼近真实值函数。从不动点准确性的角度出发,提出了基于 n 步自举法的不动点解的扩展;从投影角度出发,提出了基于线性插值法的不动点解的构造。给出了相对应的证明和分析。将广义不动点的思想应用于成熟的 CBMPI 算法框架上,提出了基于广义不动点的 CBMPI(n,β)算法。通过在 Tetris 游戏上试验发现,广义不动点算法比经典的不动点算法更好,同时还发现不同的 n 和 β 对基于广义不动点的 CBMPI(n,β)算法影响较大。

针对逼近最优解这一问题,提出了基于贝叶斯优化的广义不动点解的参数优化,搜索和优化更好的 n 和 β,期望逼近最优解,找到更好的次优解。通过在 Tetris 游戏上试验发现,贝叶斯优化方法行之有效,能够搜寻到较优的 n 和 β,这也使得广义不动点解可以更好地逼近最优解。

提出了基于广义不动点解的集成学习,包括了两种方案,即策略集成和值函数集成,以获得更高质量的解。通过在 Tetris 游戏上试验发现,基于广义不动点解的集成学习不仅可以比基于贝叶斯优化的广义不动点得分更高,还能比一些文献中记录的应用于俄罗斯方块游戏中的方法效果更好。

参考文献:

[1] 陈兴国, 孙丁源昊, 杨光, 等. 不动点视角下的强化学习算法综述[J]. 计算机学报, 2023, 46(6): 1246-1271.
CHEN Xingguo, SUN Dingyuanhao, YANG Guang, et al. A Survey of Reinforcement Learning Algorithms from a Fixed Point Perspective[J]. Chinese Journal of Computers, 2023, 46(6): 1246-1271.

[2] GEIST M, PIETQUIN O. Algorithmic survey of parametric value function approximation[J]. IEEE Transactions on Neural Networks and Learning Systems, 2013, 24(6): 845-867.

[3] PUTERMAN M L. Markov decision processes: discrete stochastic dynamic programming[M]. New York, USA: John Wiley & Sons, 2014.

[4] SUTTON R S. Learning to predict by the methods of temporal differences[J]. Machine Learning, 1988, 3(1): 9-44.

[5] BRADTKE S J, BARTO A G. Linear least-squares algorithms for temporal difference learning[J]. Machine Learning, 1996, 22(1-3): 33-57.

[6] GERAMIFARD A, BOWLING M, SUTTON R S. Incremental least-squares temporal difference learning[C]//Proceedings of the 21st National Conference on Artificial Intelligence. Massachusetts, USA: AAAI Press, 2006: 356-361.

[7] ERNST D, GEURTS P, WEHENKEL L. Tree-based batch mode reinforcement learning[J]. Machine Learning, 2005, 6(4): 503-556.

[8] KUMAR H, KOPPEL A, RIBEIRO A. On the sample complexity of actor-critic method for reinforcement learning with function approximation[J]. Machine Learning, 2023: 1-35.

[9] SUTTON R S, MAEI H, SZEPESVáRI C. A convergent O(n) temporal-difference algorithm for off-policy learning with linear function approximation[C]//Advances in Neural Information Processing Systems.Cambridge, USA: MIT Press, 2008: 1609-1616.

[10] SUTTON R S, MAEI H R, PRECUP D, et al. Fast gradient-descent methods for temporal-difference learning with linear function approximation[C]//Proceedings of the 26th International Conference on Machine Learning. New York, USA: ACM, 2009: 993-1000.

[11] BAIRD L. Residual algorithms: reinforcement learning with function approximation[C]// Proceedings of the 12th International Conference on Machine Learning. San Francisco, USA: Morgan Kaufmann Publishers Inc, 1995: 30-37.

[12] SCHERRER B. Should one compute the temporal difference fix point or minimize the bellman residual? the unified oblique projection view[C]//Proceedings of the 27th International Conference on Machine Learning. Madison, USA: Omnipress, 2010: 959-966.

[13] TSITSIKLIS J, VAN ROY B. An analysis of temporal-difference learning with function approximation technical[J]. IEEE Transactions on Automatic Control, 1997, 42(5): 674-690.

[14] BOYAN J A. Technical update: least-squares temporal difference learning[J]. Machine Learning, 2002, 49(2-3): 233-246.

[15] GERAMIFARD A, BOWLING M, ZINKEVICH M, et al. iLSTD: eligibility traces and convergence analysis[C]// Advances in Neural Information Processing Systems. Cambridge, USA: MIT press, 2006: 441-448.

[16] MAEI H R, SUTTON R S. GQ(λ): a general gradient algorithm for temporal-difference prediction learning with eligibility traces[C]//Proceedings of the 3rd Conference on Artificial General Intelligence. Paris, France: Atlantis, 2010: 91-96.

[17] JOHNS J, PETRIK M, MAHADEVAN S. Hybrid least-squares algorithms for approximate policy evaluation[J]. Machine Learning, 2009, 76(1): 243-256.

[18] 吴毓双, 陈筱语, 马静雯, 等. 基于一般化斜投影的异策略时序差分学习算法[J]. 南京大学学报(自然科学版), 2017, 53(6): 1052-1062.
WU Yushuang, CHEN Xiaoyu, MA Jingwen, et al. Off-policy linear temporal difference learning algorithms with a generalized oblique projection[J]. Journal of Nanjing University(Natural Science), 2017, 53(6): 1052-1062.

[19] SUTTON R S, BARTO A G. Reinforcement learning: an introduction[M]. Cambridge, USA: MIT press, 2018.

[20] BERTSEKAS D, TSITSIKLIS J N. Neuro-dynamic programming[M]. Belmont, USA: Athena Scientific, 1996.

[21] ANTOS A, SZEPESVÁRI C, MUNOS R. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path[J]. Machine Learning, 2008, 71(1): 89-129.

[22] SCHERRER B, GHAVAMZADEH M, GABILLON V, et al. Approximate modified policy iteration and its application to the game of Tetris[J]. Machine Learning, 2015, 16(49): 1629-1676.

[23] 厉海涛, 金光, 周经伦, 等. 贝叶斯网络推理算法综述[J]. 系统工程与电子技术, 2008, 30(5): 935-939.
LI Haitao, JIN Guang, ZHOU Jinglun, et al. A review of bayesian network inference algorithms[J]. Systems Engineering and Electronics, 2008, 30(5): 935-939.

[24] LIOR R. Ensemble-based classifiers[J]. Artificial Intelligence Review, 2010, 33(1-2): 1-39.

[25] LICHTENBERG J M, ŞIMŞEK Ö. Regularization in directable environments with application to Tetris[C]//Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA: IMLS, 2019: 3953-3962.

[26] DEMAINE E D, HOHENBERGER S, LIBEN-NOWELL D. Tetris is hard, even to approximate[C]// Proceedings of the 9th International Conference on Computing and Combinatorics. Berlin, Germany: Springer, 2003: 351-363.

[27] FARIAS V F, VAN ROY B. Tetris: A study of randomized constraint sampling[C]//Probabilistic and Randomized Methods for Design under Uncertainty. London, UK: Springer, 2006: 189-201.

[28] THIERY C, SCHERRER B. Improvements on learning Tetris with cross entropy[J]. International Computer Games Association Journal, 2009, 32(1): 23-33.

(编辑:陈燕)

(上接第 20 页)

[27] WANG X, WANG S, DU Y, et al. Minimum class variance multiple kernel learning [J]. Knowledge-Based Systems, 2020, 208(5): 106469.

[28] RAHIMI A, RECHT B. Random features for large-scale Kernel machines [EB/OL]. (2019-05-28) [2019-05-28]. https://doi.org/10.48550/arXiv.2209.01958.

[29] HAN I, AVRON H, SHIN J. Polynomial Tensor Sketch for Element-wise Function of Low-Rank Matrix[C]// Proceedings of the 37th International Conference on Machine Learning. Vienna, Austria: MIT Press, 2020: 3942-3951.

[30] CHO Y, SAUL L. Kernel methods for deep learning [C]// In Advances in Neural Information Processing Systems 22. Vancouver, Canada: Red Hook: 2009: 342-350.

[31] BIETTI A, MAIRAL J. On the inductive bias of neural tangent kernels[EB/OL]. (2019-05-29)[2019-10-31]. ttp://arxiv.org/abs/1905.12173.

[32] ZANDIEH A, HAN I, AVORN H, et al. Scaling neural tangent kernels via sketching and random features [J]. Advances in Neural Information Processing Systems, 2021, 34: 1062-1073.

[33] PHAN N, PAGH R. Fast and scalable polynomial kernels via explicit feature maps[C]//Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Chicago, USA: ACM, 2013: 239-247.

[34] AIOLLI F, DONINI M. EasyMKL: a scalable multiple kernel learning algorithm[J]. Neurocomputing, 2015, 169: 215-224.

[35] TANABE H, HO T B, Nguyen C H, et al. Simple but effective methods for combining kernels in computational biology[C]//Proceedings of the 2008 IEEE International Conference on Research, Innovation and Vision for the Future in Computing and Communication Technologies. Ho Chi Minh City, Vietnam: IEEE, 2008: 71-78.

(编辑:陈燕)