Policy Regularization for Legible Behavior

陈梦林

摘要

摘要：为了将代理的策略与其他一组策略区分开来，本文介绍了一个允许正则化强化学习代理的易读性模型，该模型提出了一个易读性标准，又通过该标准引入了一个观察者模型。在不修改智能体学习算法的前提下，将智能体和观察者建模为等价的贝叶斯网来包装 Q_learning 学习的策略集，并提出了一种专注于观察者对代理的推断的二阶心理理论模型，代理通过实现二阶心理理论模型来模拟观察者如何推断意图，增加了代理当前的策略和策略池中其他策略之间的区别，而且代理的最优策略可以通过正则化使观察者推断出错误的策略的观察结果来提高可读性。本文通过实现网格实验来评估该易读性方法，该实验为说明性实验，说明由易读性引入的决策边界会影响代理的策略返回在其他策略中也具有很高概率的动作的状态，在这种情况下需要在易读和次优之间做决策。总之，当正则化因子保持在一个合理数值时，本文的模型成功的增加了轨迹的易读性，并且不改变代理的学习过程

关键词：易读性；可解释性；

1 引言

随着科技的进步，人机合作已经机器人之间的合作变得越来越普遍，代理不仅要高效、准确地执行分配给他们的任务，还应该确保操作环境中的人和其他代理能够理解其意图以及行动，通过人类可以识别理解的行为促进策略的识别，即增加人对代理策略的可信度，可以提高合作者对机器人的信任度等。例如当餐厅服务员与机器人合作清理通一张餐桌上的脏盘子时，餐厅服务员需要根据代理当前的行为判断，代理当前想要收的是哪一个盘子，从而去处理其他的盘子，所以说代理意图的表达在人工代理的交互中变得越来越重要了，因此将意图表达的方法和产生高性能行为的技术相结合变得越来越重要。目前强化学习可以为各种领域产生强大的代理已被证明，但在强化学习中可解释的的定义无法满足在线设置，因为在在线设置中交互的流畅性阻碍了对决策算法的深入检查。所以为了满足在线设置中的可解释性，借鉴了可解释规划文献中的方法，该方法专注于代理的易读性，通过使其意图在观察者模型中容易被识别。又由于可解释性的规划文献中与强化学习框架相关的框架很少，所以将可解释性方法应用于强化学习中存在很大的未开发潜力，本文将可解释规划的易读性标准转化为 RL 框架作为策略可读的衡量标准。

2 相关工作

2.1 强化学习的可解释性定义

由于强化学习中可解释性的定义来自于机器学习中，在机器学习中可解释性通常意味着提高对代理机制的洞察，以便在专家检查时能够理解其决策[1]。这在强化学习中可以通过将分类器的潜在特征转化为一个可解释的空间来实现[2]，然后在该空间上计算解释。这些可解释性技术在许多机器学习领域被证明是有用的，可以洞察模型的决策，但同样不适合实时交互的领域，其一是交互的流畅性阻碍

了对决策算法的深入检查，其次从特征方面产生的可以被专家理解的可解释性，可能不适合于专注于常识推理的底层模型的用户。

2.2 可解释性规划中可解释定义

在文中将计划中的可解释性也称为可解释性被定义为当观察者可以通过理解代理的意图轻松识别代理正在做什么时，此时代理的行为是可解释的[3]。此定于在强化学习中同样适用，而且更加符合观察者在线交互，其中观察者可以是被动的，也可以是更大的协作代理的一部分。其中由于对于可解释性方面的捕捉，都表达的是观察者对代理的各种期望模型，例如在本文中的是对其目标的期望[4]，对整个未来轨迹的期望[5]，或对通信模型的期望[6]等等，为此本文描述了一种专注于观察者对代理的推断的二阶心理理论模型，计划中的可解释行为可被视为最小化观察者的推断模型与代理真实模型之间的距离。当代理的模型符合二阶心理理论模型时代理的策略是可解释的，否则是不可解释的。

3 本文方法

3.1 本文方法概述

本文首先定义直接适用于策略的易读性标准，即如果一个代理的策略与一组其他策略可区分，则该策略是易读的，在本文中，该定义表示为观察者试图理解代理当前在一组策略中选择了哪一个策略。代理可以通过实现二阶心理理论来模拟它推断意图的期望，为了实现二阶心里理论模型，本文提出了镜像代理模型下图 1 为镜像代理模型，它将代理模型和二阶心理理论模型作为等效的贝叶斯网，分别为 P_R 和 P_R^H。其中 P_R 为代理模型，决定代理然后行动；P_R^H 为观察者模型，是观察者关于如何推导代理的模型。由于代理的推导和观察者的先验信息不一样所以随机变量 (Π,S 和 A) 在 P_R 和 P_R^H 的分布也不同。假设代理有一组固定的预训练策略集为 $(\pi_0, \pi_1, \pi_2 ... \pi_n)$, 由于策略为一系列动作的集合当代理的策略确定时，代理每个状态对应的动作也就确定了即 $P_R(\pi = \pi_R) = 1$，又由于观察者开始被建模为不知道代理正在执行哪一个策略，所以导致观察者关于策略的一致先验概率为 $P_R^H(\pi_R) = k = \frac{1}{|\pi|}$。又根据 Q_Learning 得到了 Q 值，根据 Q 值通过贪婪策略选择代理每一个状态的 Q 值最大的动作，代理和观察者的 Q 值分别用 $Q_R(A, \pi, S)$ 和 $Q_R^H(A, \pi, S)$，其中 $P_R(A|\pi, S) = f(Q_R(A, \pi, S))$ 和 $P_R^H(A|\pi, S) = g(Q_R^H(A, \pi, S))$ 分别确定代理选择动作的概率分布以及观察者推断动作的概率分布，其中 f 和 g 通过 Boltzmann 探索将 Q 值转换为动作概率的分布函数[7]，其中 Boltzmann 探索为一种模仿 softmax 为不同动作的 Q 值进行规范化又称归一化的函数。

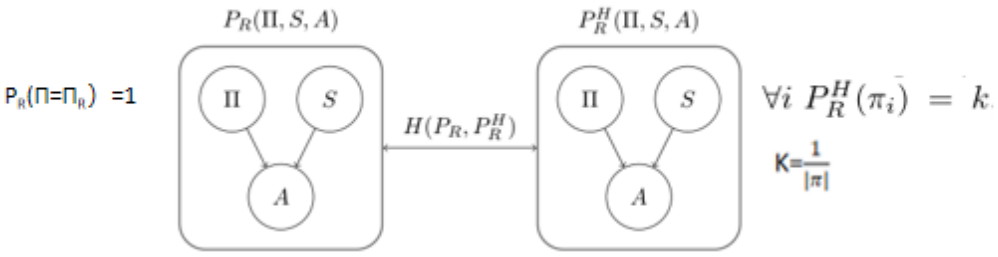

图 1: 镜像代理模型图

3.2 Q_learning 模块

如何得到代理以及观察者的策略？策略本质上为一系列相关动作的集合，以达到一个最大的 reward 值，本文主要通过 Q_Learning 得到了 Q 值，然后根据 $\in -greedy$ 策略选择代理每一个状态的 Q

值最大的动作，并更新 Q 值表。通过 Q_learning 学习 70 万次得到 Q 值表，其中训练的部分参数为：discount=0.9，learning_rate=0.1,policy_epsilon=0.4 网格大小为 20*20。Q_learning 采用异策略时间差分方法来实现该算法，接下来我主要从 Q_learning 的伪代码来对 Q_learning 算法进行推导如图 2 所示

1.	初始化 Q(s,a),∀ s ∈ S,a ∈ A(s),给定参数 学习率α,折扣系数γ,探索率ϵ
2.	Repeat:
3.	给定起始状态 s，并根据ϵ贪婪策略在状态 s 下选择动作
4.	Repeat()对于每一个 episode：
5.	（a）根据ϵ贪婪策略在状态S_t，选择动作a_t，得到回报R_t和下一个状态S_{t+1}
6.	（b）$Q(s_t,a_t)\longleftarrow Q(s_t,a_t)+\alpha[R_t+\gamma max_{a'}Q(s',a')-Q(s_t,a_t)]$
7.	（c）$s\longleftarrow s'$
8.	Until s 是终止状态
9.	Until 所有的 Q(s,a)收敛
10	输出最终策略：$\pi(s)=\mathrm{argmax}_a Q(s,a)$

图 2: Qlearning 推导图

Qlearning 算法是异策略时间差分方法，其中异策略指他的行动策略（即产生数据的策略）和要评估的策略不是一个策略，在图中 Qlearning 伪代码中，行动策略是第 5 行的 $\in -greedy$ 其中在本文中 $\in=0.4$，而要评估和改进的策略是第 6 行的贪婪策略；时间差分方法，是指利用时间差分目标来更新当前行为值函数，在上图中的第 6 行中时间差分目标为 $R_t+\gamma max_{a'}Q(s',a')$。

3.3 交叉熵函数定义

为了可读性，代理应该选择动作向观察者传递其目前正在执行的策略 π_R，可以通过如何减小代理策略上的概率分布 $P_R(\pi)$ 和观察者根据状态和动作推断的概率公布 $P_R^H(\pi|S,A)$ 之间的距离来选择动作。下面是交叉熵公式：

$$\begin{aligned} H(P_R(\Pi),P_R^H(\Pi|s,a)) &= \\ -\log P_R^H(\pi_R|a,s) &= \\ -\log P_R^H(a|\pi_R,s)+\log\mathbb{E}[P_R^H(a|\pi,s)]-\log P_R^H(\pi_R). & \end{aligned}$$

根据每一个状态的最小交叉熵选择动作，如何又根据得到的最小交叉熵来调整 Q 值，以此来得到正则化后的 Q 值，Q 值的正则化版本：

$$\begin{aligned} Q_{\text{leg}}(\pi_R,s,a) &= \\ Q_R(\pi_R,s,a)-\alpha H(P_R(\Pi),P_R^H(\Pi|s,a)) &= \\ Q_R(\pi_R,s,a)+\alpha\log P_R^H(\pi_R|a,s)). & \end{aligned}$$

其中 $\alpha>0$ 表示正则化的大小，如上公式因为在 Q_Learning 中动作的选择取决于 Q 值，所以正则化了 Q 值表可以等价于正则化了策略，如上式，正则化后的选择的操作组成的策略与观察者推断的策略之间的距离很小。又由 Q 值正则化版本所示由于易读性的引入影响了决策边界，返回在其他策略中也具有很高概率的动作的状态，在这种情况下需要在易读和次优之间做决策。

4 复现细节

4.1 与已有开源代码对比

本文在第二个 DQN 实验的时候找了作者要了源码，但由于操作系统以及 python 等版本的原因，代码无法正常运行，由于第一次实现此类型的论文，开始比较迷茫，不知道如何上手，后面参照作者源码，改写自己的代码，进行一系列调参，成功实现了第一个实验，至于第二个实验，该实验为 DQN 实验是 Q_learning 与神经网络的结合，根据作者提供的游戏代码无法实现环境的注册，然后自己在网上查找类似的方法进行多次实验成功将游戏环境注册成功，然后开始神经网络的构建，由于源码中网络参数的错误和论文中没有提供卷积层等一系列信息，更加由于自己对神经网络知识的不精通，导致一直没有运行成功，第二个实验复现失败。

4.2 实验环境搭建

为了方便进行环境的管理，在本次复现时，下载安装了 anaconda，它是一个开源的包、环境管理器，其中包含了 conda、python180 多个科学包及其依赖项，且以及安装好了实验是用到的 numpy,pandas 等库，可以用于在同一个机器上安装不同版本的软件包及其依赖，并可以建立多个虚拟环境，用来隔离不同项目所需的不同版本的工具包，以防止版本上的冲突。

4.3 创新点

本文由于自身时间以及知识储备有限暂时只做到了复现，没有创新。

5 实验结果分析

本部分主要对实验所得结果进行分析，说明实验内容进行说明，论文中的实验结果如下图 2 所示，复现的实验结果如图 3 所示。

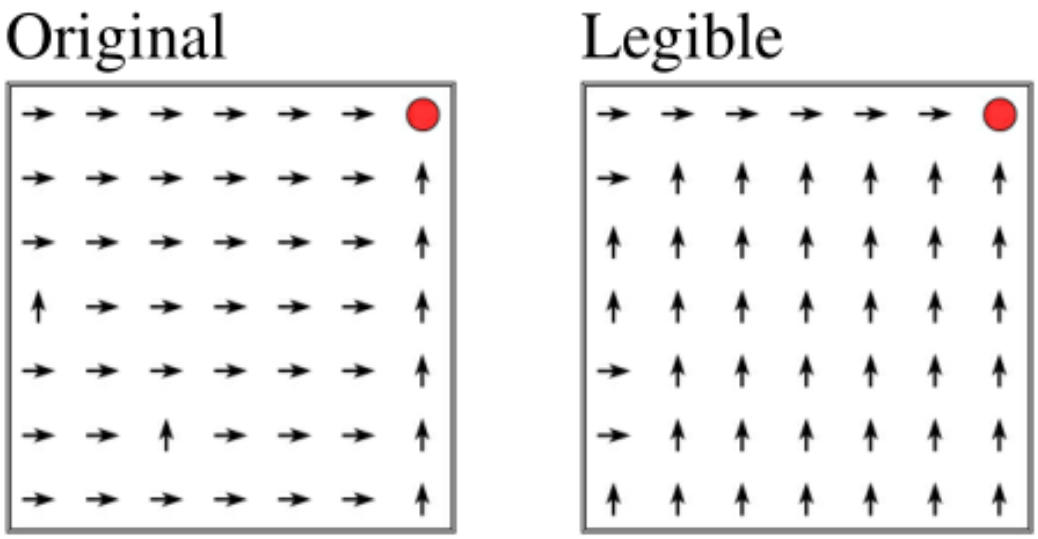

图 3: 论文中的实验结果示意

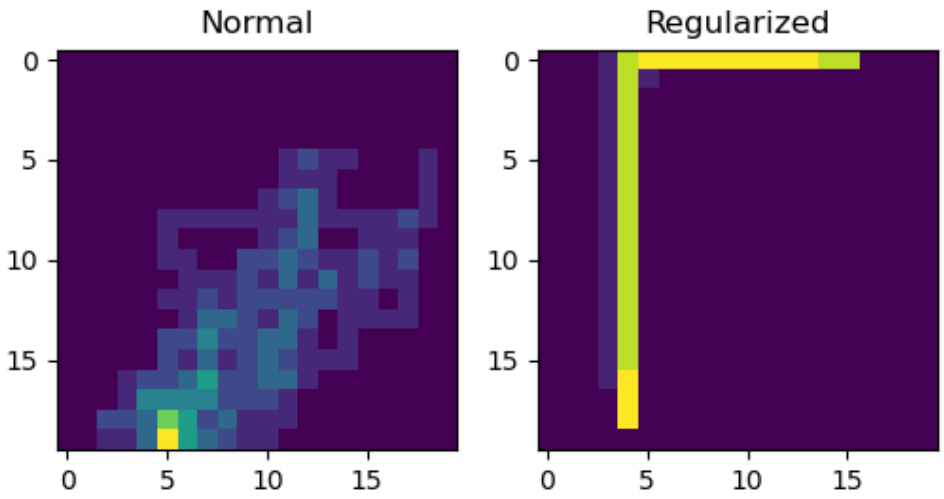

图 4: 实验结果示意

为了简化实验本文在用 Q_learning 训练时，将代理的 Q 值以及观察者的 Q 值设置为一样以及将 Q 值转化为动作概率的分布函数 f 和 g，这意味着代理假设观察者使用与自己相同的 q 值和派生的动作概率，即 $\forall_i P_R(A|\pi_i, S) = P_R^H(A|\pi_i, S)$ 代理与观察者选择动作的概率相等。如图所示在右列中，使用 α = 1 获得的相应清晰的策略。学习到的策略移动到目标附近的一堵墙，然后通过沿着墙走来接近目标。然而，为了清晰，接近正确的墙壁以消除目标位置的歧义是很重要的。清晰的政策系统地接近一堵明确的墙。

6 总结与展望

本文通过实现网格实验来评估该易读性方法，该实验为说明性实验，说明由易读性引入的决策边界会影响代理的策略返回在其他策略中也具有很高概率的动作的状态，在这种情况下需要在易读和次优之间做决策。总之，当正则化因子保持在一个合理数值时，本文的模型成功的增加了轨迹的易读性，并且不改变代理的学习过程。由于自身能力的原因没有将第二个实验复现出来，我将继续学习争取早日复现出来。

参考文献

[1] DU M, LIU N, HU X. Techniques for Interpretable Machine Learning[J]., 2018.

[2] ROSCHER R, BOHN B, DUARTE M F, et al. Explainable Machine Learning for Scientific Insights and Discoveries[J]. IEEE Access, 2020, PP(99): 1-1.

[3] CHAKRABORTI T, KULKARNI A, SREEDHARAN S, et al. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable Agent Behavior[C]//. 2018.

[4] DRAGAN A D, LEE K C T, SRINIVASA S S. Legibility and Predictability of Robot Motion[J]., 2013.

[5] ZHANG Y, SREEDHARAN S, KULKARNI A, et al. Plan Explicability and Predictability for Robot Task Planning[Z]. 2015.

[6] MACNALLY A M, LIPOVETZKY N, RAM#XED M, et al. Action Selection for Transparent Planning

[J]., 2018.

[7] SZEPESVÁRI C. Synthesis Lectures on Artificial Intelligence and Machine Learning[M]. Synthesis lectures on artificial intelligence, 2009.