RUDDER - Reinforcement Learning with Delayed Rewards — hoho

论文试图解决什么问题？

解决Delayed reward问题，从而可以有效的、直接的对reward产生的状态-动作对（state-action pairs）进行credit

现有方法的问题：

1. Do not use temporal difference (TD) since it suffers from vanishing information and leads to high bias, even with eligibility traces.
2. Do not use Monte Carlo (MC) since averaging over all possible futures leads to high variance.
3. Do not use conventional value functions based on state-action pairs since the expected return has to be predicted from every state-action pair. A single prediction error might hamper learning.

这是否是一个新的问题？

否

这篇文章要验证一个什么科学假设？

奖励是可以通过分解（Reward Decompostion）从而进行重分配（Reward Redistribution）

有哪些相关研究？如何归类？谁是这一课题在领域内值得关注的研究员？

todo

论文中提到的解决方案之关键是什么？

1. RUDDER的目标是区别于传统的RL方法，要构建一个使得未来期望回报等于0的MDP，使得Q值估计可以简化为只计算立即回报的期望，因为这可以简化Q函数的估计：根据贝尔曼方程，Q函数为$q^{\pi}\left(s_t, a_t\right) = r\left(s_t, a_t\right) + \kappa\left(T-t-1, t\right)$，其中：
   - $\kappa\left(m, t-1\right)$为在t-1时刻其后面m步的期望未来回报，即从$R_{t+1}$到$R_{t+1+m}$的未来奖励之和（假设折扣因子为0）
   - $r\left(s_t, a_t\right)$为t时刻的立即奖励R_{t+1}的期望

   为什么使未来的预期回报等于零可以简化Q值的估计？

   因为它减少了 Q 值估计值中的偏差。在传统的强化学习中，Q 值被定义为未来折扣奖励的期望总和。但是，当奖励延迟时，估算未来的预期回报变得更加困难，这会导致Q值估计出现偏差（使用TD学习使奖励传递呈现指数式衰减，从而导致估算偏差，而使用MC学习又回导致估算出现高方差）。通过将期望的未来奖励设为零，Q 值估计值就等于计算即时奖励的平均值，从而减少了 Q 值估计值中的偏差。偏差的减少可以更准确地估计 Q 值，这对于有效的强化学习至关重要。

   最佳奖励再分配旨在实现等于零的未来预期奖励。它是指奖励的再分配，其方式是使用相同的最优策略实现回报等效的决策过程，而在最佳情况下，预期的未来奖励为零。换句话说，这是一种修改原始马尔可夫决策流程（MDP）的方法，方法是重新分配奖励，使预期的未来奖励为零，同时保留最优策略。

2. 通过贡献分析进行奖励分解

   模式识别可以解决奖励分解问题。有哪些行为或状态会导致奖励变高或变低？这些可以通过模式识别进行。

监督学习可以确定这些状态动作的模式并将奖励归因于这些事件，于是RUDDER尝试通过监督学习识别这些关键state-action从而将奖励重分配给它们。

文中提出一些概念：

- 序列马尔可夫决策过程（SDP）: 与MDP类似，但不要求奖励具有马尔可夫性
- 等价回报决策过程（Return-equivalent decision processes）：两个决策过程回报等价，当满足 : 1. 它们仅仅在奖励转移分布上有所不同；2. 对于所有的策略它们在t=0具有相同的期望回报；

由此可见，RUDDER就是要将原来的MDP转化为一个回报等价但不同（延迟奖励更少）的SDP，因为这更容易学习。

奖励分解决定每一对状态动作对回报的贡献度的问题，这是通过一个贡献度函数预测每条状态动作序列对应的回报实现的。概念上讲，本文通过一种贡献度分析方法：预测差异）Differences in prediction）来实现。

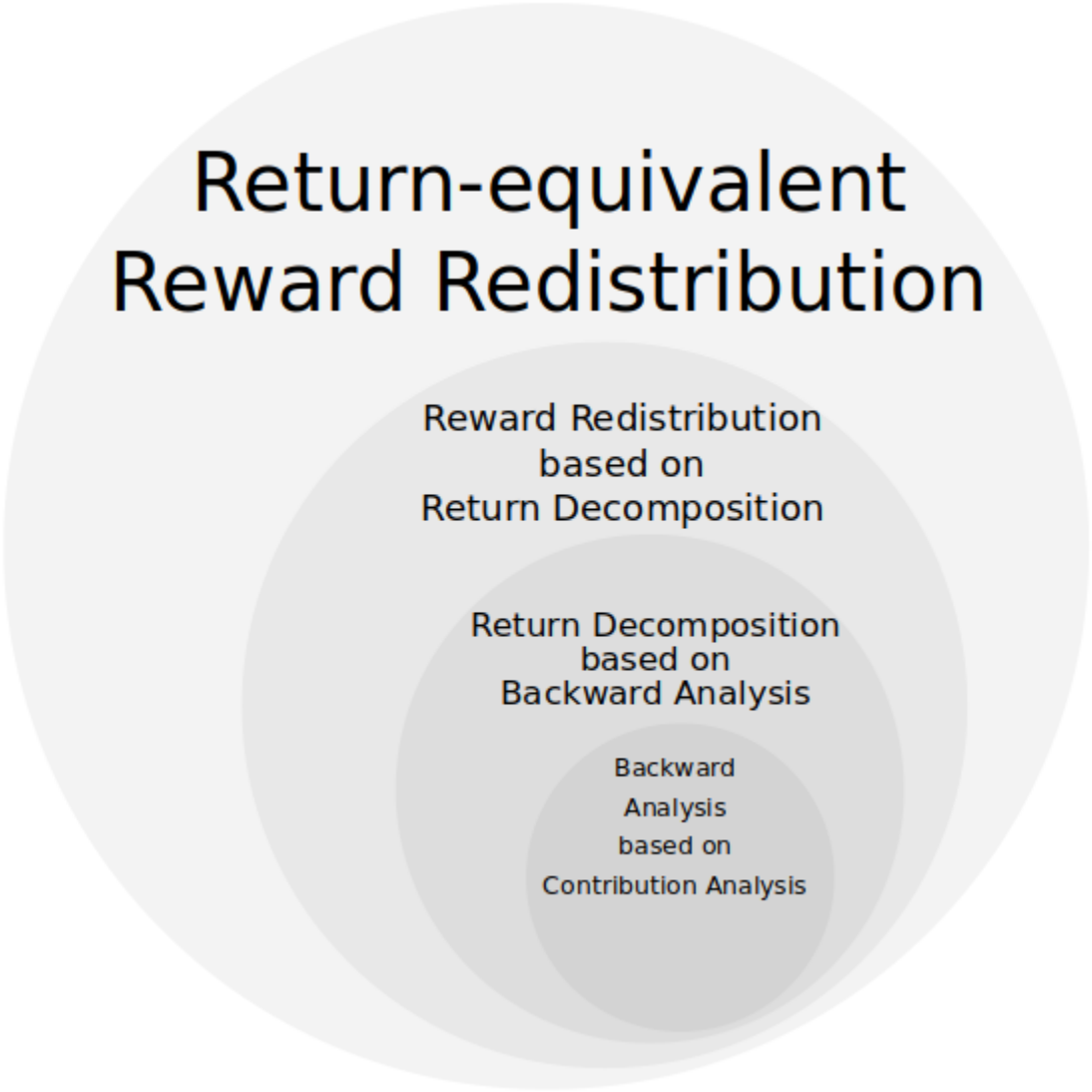
预测差异用于将总体回报分解为连续时间步长预测回报之间的差异之和。利用这些差异来构建奖励再分配，从而实现回报等效的决策过程，而预期的未来奖励为零。

预测差异背后的原理是，两个连续时间步的预测回报之间的差异代表了在中间时间步长中采取的行动的贡献。通过计算所有时间步长的这些差异，我们可以衡量每个动作对总体回报的贡献。然后，该措施可用于重新分配奖励，从而消除延迟奖励，从而实现回报等效的决策过程，未来预期奖励为零。

本文建议使用长短期记忆（LSTM）网络来预测每个时间步的全序列回报，然后使用预测回报之间的差异来构造奖励再分配。

（作者也提到了另外两种贡献分析方法：Integrated gradients、Layer-wise relevance propagation，但首选Differences in prediction）

层层推导：



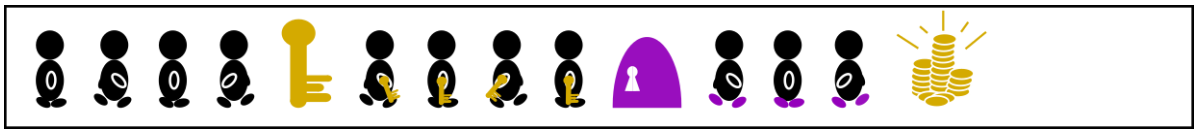
3. 使用LSTM实现奖励重分配

可以将价值函数看成是一个“阶梯函数”（step function）。

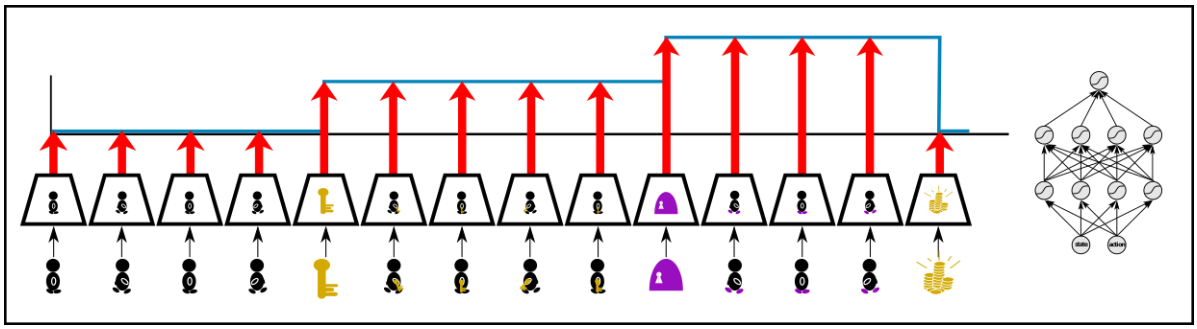
因为复杂的任务可以分层为一些子任务或子目标，它们可看作是一个个层次（step），所以，”阶梯“反映了任务的成果、或者环境的变化。

因此，识别这些阶段（上面提到的通过监督学习进行模式识别），将奖励重新分配给这些阶段很重要。

如key-door实验：



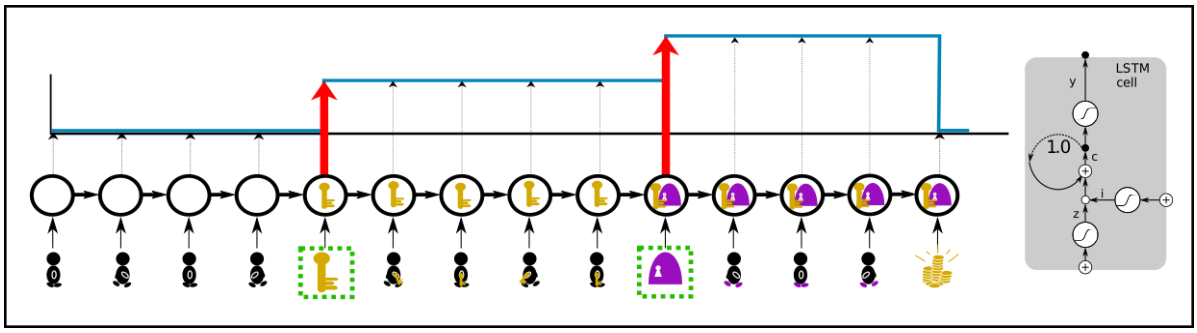
拿到钥匙和开门都可以增加获得财宝的机会，因此，我们要求解的价值函数是有”获取钥匙“和”开门“这两个重要阶段：



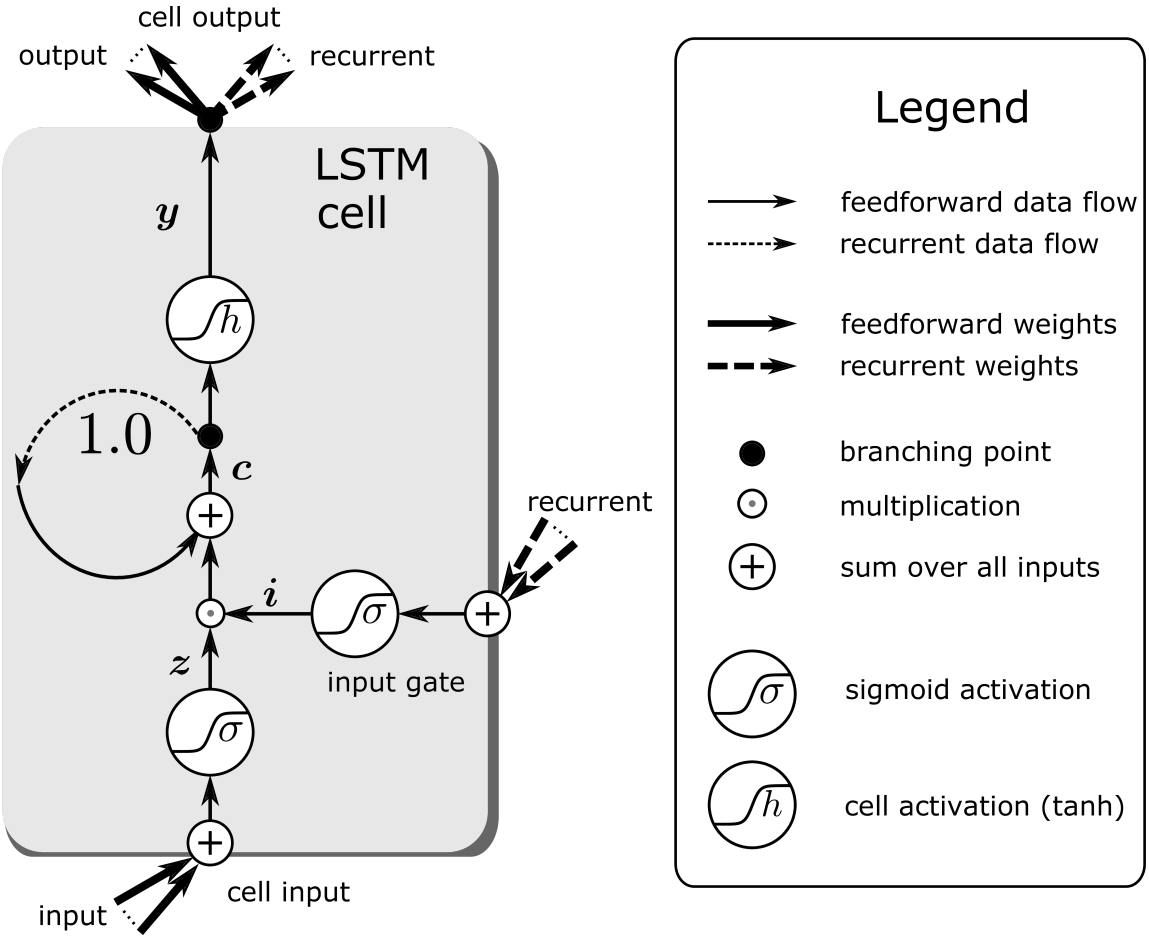
- 为什么用通常的神经网络不能通过state-action pairs学习这种阶梯函数(step function)？

  网络的学习需要每个state-action pairs以及与之对应的期望奖励，但是，从key-door实验这种环境来看，这种期望回报从获得钥匙到即将开门之前这段期间不会变化，类似的，这种期望回报从开了门到最后即将获得财报这段期间也不会变化。（However, the expected return does not change after “getting the key” until “opening the door” and after “opening the door” until “getting the treasure”. Consequently, in these intervals the value function is constant. There is no new information regarding the expected return, whereas only the previous information is repeatedly extracted.）

而LSTM等循环神经网络可以解决普通神经网络的问题。

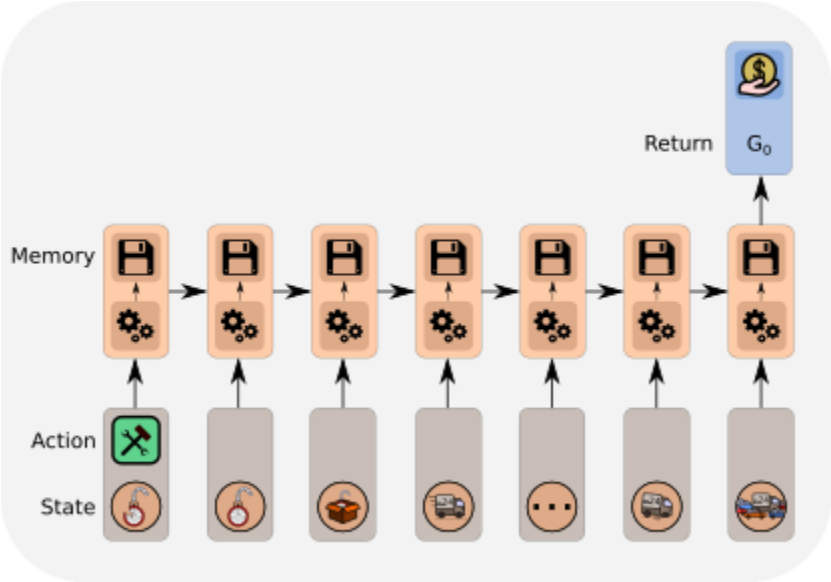


得益于LSTM的结构：



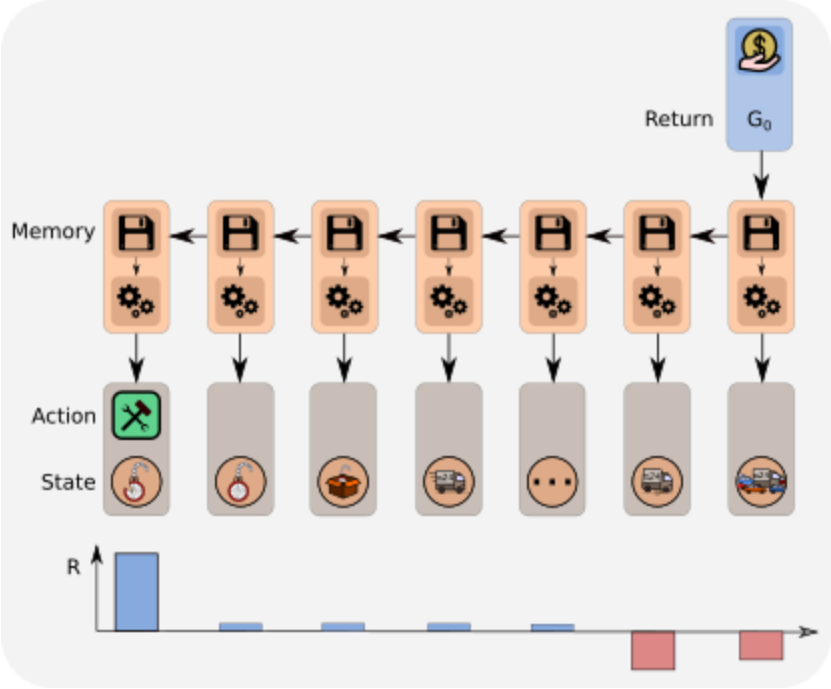
如果信息没变化网络不会学到新的东西，当有新的信息（获得钥匙和开门），相应的模式就会被学习到（就是上面提到的通过监督学习进行模式识别）（Only the patterns of the relevant events “getting the key” and “opening the door” have to be

learned to represent the whole value function…. for irrelevant events the LSTM memory cells do not change and the LSTM output remains constant as desired）

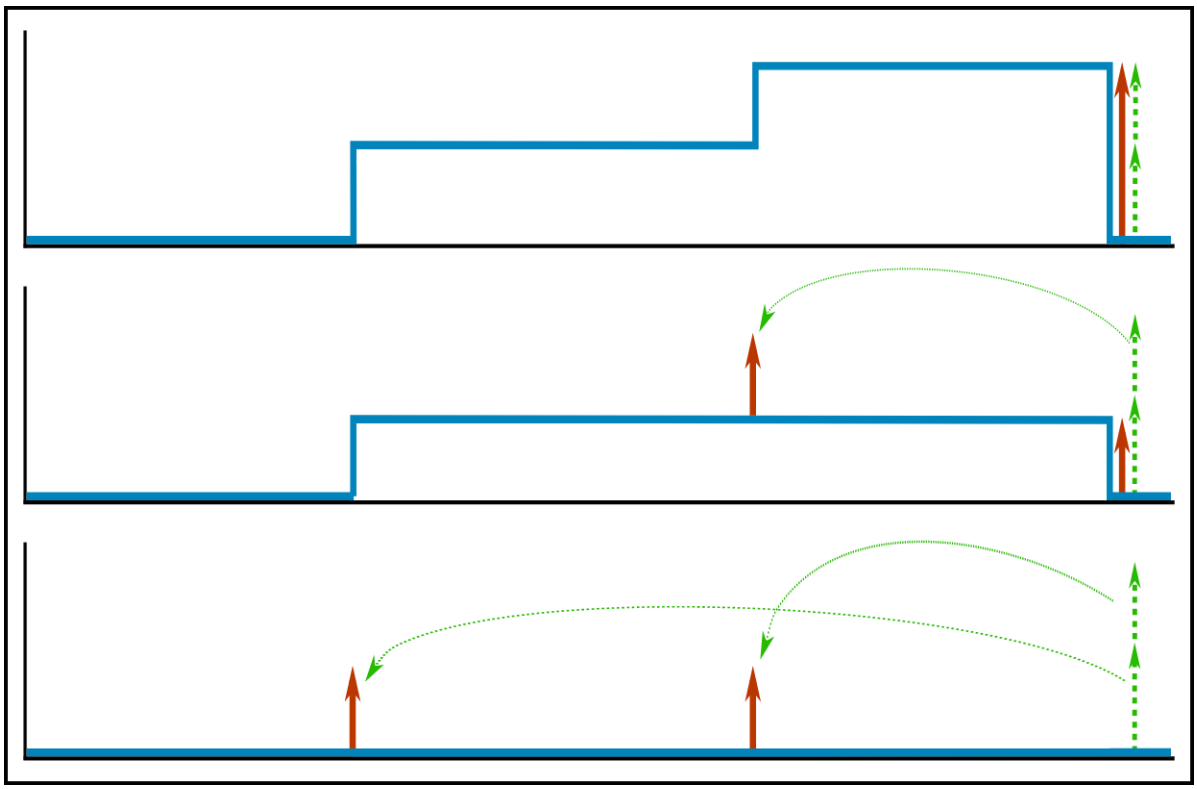


LSTM是怎么实现奖励重分配的？

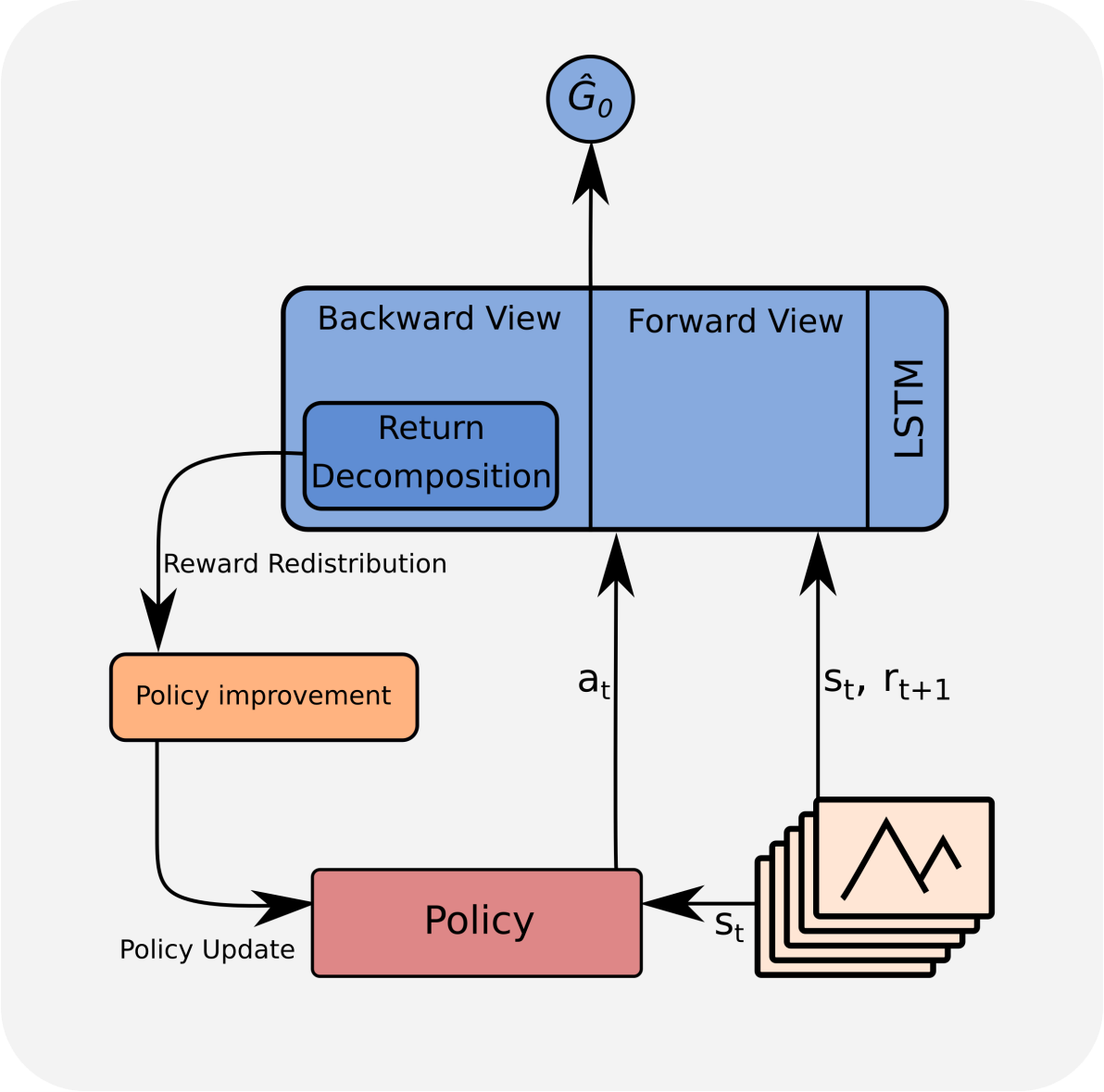
通过分析LSTM的“记忆”，可以重构出关键事件这些信息，并对最后预测的奖励的贡献度分配给每个states-action对：



通过LSTM，奖励分解可以识别出阶梯函数的每个step（如下图绿色箭头），重分配的奖励会用来替换原有的奖励。



总体流程如下图：



LSTM可以组合任何的RL算法，进行奖励重分配（如PPO）

论文中的实验是如何设计的？

todo

用于定量评估的数据集是什么？代码有没有开源？

代码：https://github.com/ml-jku/rudder

论文中的实验及结果有没有很好地支持需要验证的科学假设？

todo

这篇论文到底有什么贡献？

todo

下一步呢？有什么工作可以继续深入？

todo