Improving Reward-Based Hindsight Credit Assignment

Aditya A. Ramesh[1] **Jiamin He**[2] **Jürgen Schmidhuber**[3,1] **Martha White**[2,4]
[1]The Swiss AI Lab IDSIA, USI & SUPSI
[2]University of Alberta and the Alberta Machine Intelligence Institute (Amii)
[3]AI Initiative, King Abdullah University of Science and Technology (KAUST)
[4]Canada CIFAR AI Chair

Abstract

Accurately attributing credit or blame for outcomes to past actions is crucial for sample-efficient reinforcement learning. While temporal-difference learning with λ-returns is the most commonly used approach, it attributes credit based on the temporal proximity of actions and outcomes–a heuristic that may be overly simplistic in complex environments. Hindsight-based approaches offer an alternative by using a model that leverages future information to more explicitly credit previous actions that were critical to achieving specific outcomes. Recent work has shown that predicting past actions using future rewards can be effective for hindsight credit assignment in Markovian environments. However, we show that the associated credit-assignment algorithm suboptimally handles immediate rewards, potentially resulting in high variance even with perfect hindsight models. We introduce a simple correction that resolves this issue.

1 Introduction

Building on the principles of dynamic programming (Bellman, 1954), temporal-difference (TD) learning has become a central component of modern reinforcement learning (RL). Methods such as λ-returns (Watkins, 1989) and its online counterpart, TD(λ) (Sutton, 1988), form the basis for credit assignment in many widely adopted RL algorithms.

Although these methods are simple and reliable, they rely on the temporal proximity of actions and outcomes for credit assignment (Daley et al., 2024), which can hinder their ability to efficiently solve certain challenging credit assignment problems (Pignatelli et al., 2024). As alternatives to λ-return-based credit assignment, hindsight-based approaches have been proposed, including Upside-Down RL (Schmidhuber, 2019) and Hindsight Credit Assignment (HCA) (Harutyunyan et al., 2019).

In HCA, interaction sequences are stored in memory (up to some maximum context length), and the agent learns a hindsight model to directly associate rewards with state-action pairs in the context. This enables the agent to explicitly form associations across distant transitions without incurring an exponential decay like with λ-returns (Arjona-Medina et al., 2019).

With the proper setup, it may be possible that HCA presents advantages over λ-return-based temporal difference, similar to some benefits that attention-based architectures (e.g., Schmidhuber (1992); Bahdanau et al. (2015); Vaswani et al. (2017)) provide over standard recurrent neural networks (RNNs) for sequence modeling. For instance, consider a dataset consisting of sequences of the following form:

18th European Workshop on Reinforcement Learning (EWRL 2025).

$$\begin{aligned}&a, x_1^1, x_2^1, x_3^1 \dots x_n^1, 1\\&a, x_1^2, x_2^2, x_3^2 \dots x_n^2, 1\\&b, x_1^3, x_2^3, x_3^3 \dots x_n^3, 0\\&b, x_1^4, x_2^4, x_3^4 \dots x_n^4, 0\\&\dots,\end{aligned}$$

where x_i^j are sampled from some noisy distribution over tokens. A transformer could skip over the noisy inputs to associate the final value of 1 to the token a in the first position (similarly for 0 and b). In contrast, a vanilla RNN would struggle due to the long and noisy delay between the crucial token and the output value.

Similar to attention, some hindsight-based credit assignment approaches, like State-HCA (Harutyunyan et al., 2019), can estimate action-values by considering the pairwise relationships between all time steps in a sequence. State-HCA relies on learning and using a hindsight model – a type of multi-step inverse-dynamics model– to facilitate credit assignment. The hindsight model predicts a distribution over actions in a particular state using additional information, such as a future state. The predicted distribution from the hindsight model is used to re-weight the relevance of an action towards achieving a particular reward.

Meulemans et al. (2023) show that using future states for hindsight (as in state-HCA) can cause the algorithm to degenerate into the high-variance REINFORCE estimator (Williams, 1987), which uses the likelihood-ratio method (Glynn, 1990) to approximate stochastic gradient descent (Robbins and Monro, 1951; Amari, 1967). To fix this, they suggest using future rewards instead of future states. They also propose a policy gradient estimator, Counterfactual Contribution Analysis (COCOA), that removes the need for learning a reward model, which was required by the State-HCA algorithm introduced by Harutyunyan et al. (2019).

In this paper, we show that the variance of the COCOA policy gradient estimator can be excessively high even with the perfect hindsight model. The higher variance is a consequence of the different treatment of immediate rewards compared to later rewards. We propose a modified algorithm, Reward-HCA (R-HCA) that resolves this issue by showing that it is also possible to incorporate the immediate rewards with hindsight re-weighting.

2 Preliminaries

Consider the standard Markov Decision Process (MDP) formulation with state space $\mathcal{S}$, action space $\mathcal{A}$, reward space $\mathcal{R}$, and discount factor γ. At time step $t \in \{0, 1, 2 \dots\}$, the agent observes state $S_t \in \mathcal{S}$, takes action $A_t \in \mathcal{A}$ and receives reward $R_t \in \mathcal{R}$.

The agent acts with a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maps states to distributions over actions. Starting from the initial state S_0, the agent interacts with the MDP to generate a trajectory $\tau = (S_0, A_0, R_0, S_1, A_1, R_1, \dots)$. We denote the distribution over trajectories starting from state s and acting with policy π by $\mathcal{T}(s, \pi)$. Similarly, we denote the distribution over trajectories starting from (s, a) and acting with policy π by $\mathcal{T}(s, a, \pi)$.

The discounted return of a trajectory τ is given by $Z(\tau) = \sum_{t \geq 0} \gamma^t R_t$. The state value $V_\pi(s) = \mathbb{E}_{\tau \sim \mathcal{T}(s,\pi)} [Z(\tau)]$ represents the expected return from state s when following policy π. Similarly, the action value $Q_\pi(s, a) = \mathbb{E}_{\tau \sim \mathcal{T}(s,a,\pi)} [Z(\tau)]$ represents the expected return when starting from state s, taking action a and then following policy π. The advantage is defined as $A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s)$.

2.1 Advantage Actor-Critic

In the actor-critic framework, the policy (actor) is updated via a discounted approximation to the policy gradient. The update aims to improve the policy by taking a gradient step to increase the generalized advantage estimate (Schulman et al., 2015), $\hat{A}_t^{GAE(\gamma,\lambda)}$,

$$\hat{\nabla} V_\pi(S_0) = \sum_{t \geq 0} \nabla_\theta \log \pi_\theta(A_t | S_t) \hat{A}_t^{GAE(\gamma,\lambda)}. \quad (1)$$

The advantages are estimated using a learned value function or critic $V : \mathcal{S} \to \mathbb{R}$. Let $\delta_t = R_t + \gamma V(S_{t+1}) - V(S_t)$, then

$$\hat{A}_t^{GAE(\gamma,\lambda)} = \sum_{l \geq 0} (\gamma\lambda)^l \delta_{t+l}.$$

The critic is trained with temporal-difference using λ-return targets (Sutton et al., 1998). Such advantage estimates are commonly used in on-policy actor-critic approaches, including algorithms like A2C (Mnih et al., 2016) and PPO (Schulman et al., 2017).

2.2 Hindsight Credit Assignment

Hindsight Credit Assignment (HCA) uses a learned hindsight distribution to estimate targets for policy updates. Here we describe a generalization of the original State-HCA (Harutyunyan et al., 2019) that incorporates rewarding outcomes, as presented by Meulemans et al. (2023).

HCA-like approaches rely on a hindsight distribution $h_k^\pi(a|s,u)$, which encodes the probability that the agent chose action a in state s given that it observes a future rewarding outcome u, k steps after observing s, while following π. As noted in previous work, using the hindsight distribution, the action-value function can be re-expressed as

$$Q_\pi(s,a) = r(s,a) + \mathbb{E}_{\tau \sim \mathcal{T}(s,\pi)} \left[\sum_{k \geq 1} \left(\frac{h_k^\pi(a|s,U_k)}{\pi(a|s)} \right) \gamma^k R_k \right]. \tag{2}$$

The outcome U_k can take different forms. In the original state-HCA formulation, $U_k = S_k$. Meulemans et al. (2023) propose a general form of $U_k = f(S_k, A_k, R_k)$ that can be used to reduce the variance of the policy gradient estimator. As a specific instance, they introduce COCOA-reward, where $U_k = R_k$.

We approximate the hindsight distribution using a learned model $\hat{h}_k(a|s,\pi,u)$. With this learned hindsight model and a reward model $\hat{r}(s,a)$, one could use an all-action policy gradient to update the actor's parameters, θ,

$$\hat{\nabla} V_\pi(S_0) = \sum_{t \geq 0} \sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a|S_t) \left[\hat{r}(S_t,a) + \sum_{k \geq 1} \left(\frac{\hat{h}_k^\pi(a|S_t,U_{t+k})}{\pi_\theta(a|S_t)} \right)_{sg} \gamma^k R_{t+k} \right], \tag{3}$$

where the $(\cdot)_{sg}$ notation is used to denote the stop gradient operation or 'detaching' from the computation graph.

To avoid the computational overhead of learning a reward model, Meulemans et al. (2023) propose the following policy gradient update:

$$\hat{\nabla} V_\pi(S_0) = \sum_{t \geq 0} \nabla_\theta \log \pi_\theta(A_t|S_t) R_t + \sum_{a \in \mathcal{A}} \nabla_\theta \pi(a|S_t) \sum_{k \geq 1} \left(\frac{\hat{h}_k^\pi(a|S_t,U_{t+k})}{\pi_\theta(a|S_t)} \right)_{sg} \gamma^k R_{t+k}. \tag{4}$$

This update reduces to REINFORCE in the special case where $\hat{h}_k^\pi(a|S_t,U_{t+k}) = \mathbb{I}(A_t)$, where $I(\cdot)$ the indicator function placing all probability mass on the action that was actually taken, A_t.

Learning the hindsight model The success of HCA-like approaches depends on learning an accurate hindsight model ($\hat{h}_k^\pi$). Training this model is a supervised learning task that maximizes the likelihood of observed actions given states and future outcomes. Inaccurate hindsight models may be the major factor limiting the broader applicability of these methods (Alipov et al., 2021).

Learning an accurate hindsight model presents significant challenges. Since even the true hindsight distribution is policy-dependent, the hindsight model must be continuously updated as the policy changes. As the policy evolves, it introduces non-stationary action targets for state-future information pairs, complicating the learning process. Due to these reasons, hindsight models are typically trained with the latest interaction data and cannot reliably leverage replay buffers.

To address the challenges of policy dependence, previous work introduces the concept of a policy prior (Alipov et al., 2021; Meulemans et al., 2023). Rather than predicting the full hindsight distribution, the hindsight model learns only the correction or difference relative to the current policy's outputs. This enables more stable learning of hindsight models as policies evolve during training, and it provides a principled baseline prediction for past actions, one that does not rely on future information.

3 Illustrating the problem

Hindsight-based approaches are often motivated by examples like the one presented in our introduction – where one early situation is crucial to achieving a positive or negative outcome, with noise in the middle. This feature is also exemplified in the *umbrella problem*, which is commonly used to evaluate credit assignment capabilities (Osband et al., 2019).

In this section, we consider an episodic Markov decision process (MDP) based on the umbrella problem. We show that COCOA struggles in this environment even when using predictions from the ground-truth/perfect hindsight model. The failure can be attributed to the fact that the COCOA update (see update in Equation 4) does not reduce the variance associated with the immediate rewards.

Environment details The environment is parameterized by horizon H. The agent has two available actions at each time step, a_0 and a_1. Actions do not have any influence after the first time step. The action taken in the first step is reflected in subsequent states via Boolean indicators.

Concretely, the state s_t consists of 5 components $s_t^1, s_t^2, s_t^3, s_t^4, s_t^5$. The first component s_t^1 is a continuous variable indicating the time remaining; at time step t, it is $1 - t/H$. The next two components, s_t^2, s_t^3 are Booleans that indicate context, which is set with first state, i.e., every time the environment is reset. With 50% chance $s_0^2 = 1$ or $s_0^2 = 0$, s_t^3 is the Boolean not applied to s_t^2. These values remain fixed for the entire episode. The following two components are also a pair of Booleans. These are initially zero $s_0^4 = 0, s_0^5 = 0$. After the first step, they are set to $s_t^4 = 1, s_t^5 = 0$ if the first action was a_0, otherwise $s_t^4 = 0, s_t^5 = 1$.

At every time step other than the final one, the agent receives a reward of $-2/H$ or $3/H$ with uniform probability. On the final step, if the context matches the result of the first action (if $s_H^2 = s_H^4$), the agent receives a reward of $1/H$. Otherwise, the agent receives a reward of $-1/H$. The environment is depicted in Figure 1.

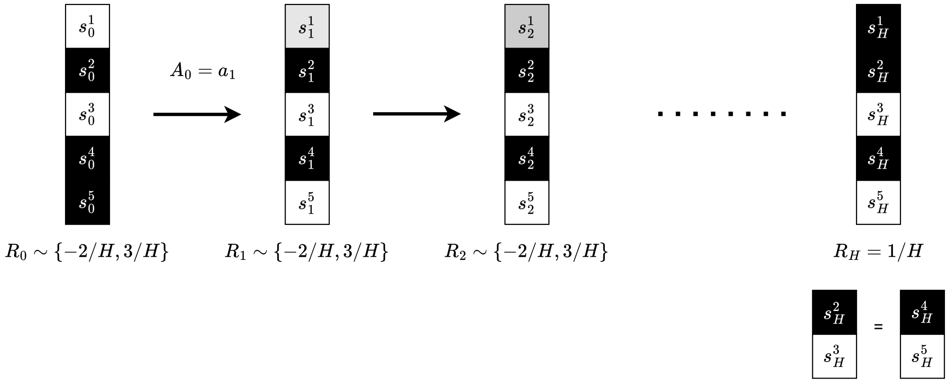

Figure 1: Illustration of the environment considered in Section 3. Black indicates the state component is zero, and white indicates that it is one. The agent receives noisy rewards on all steps apart from the final one. In the final step, the agent receives a positive reward if the context matches the consequence of the first action.

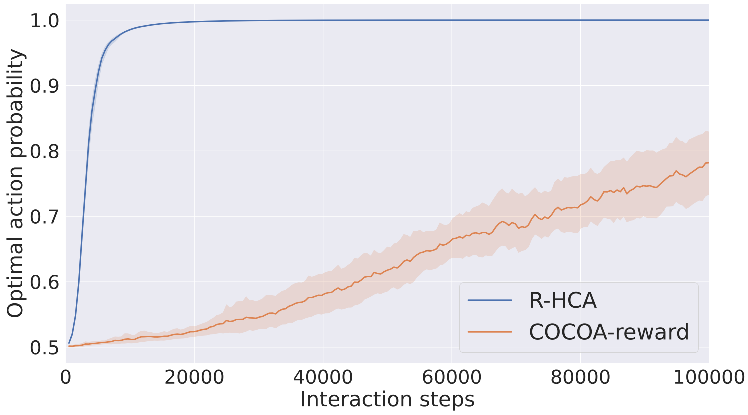

Figure 2: The probability that the policy assigns to the optimal action on the first time step in the Umbrella environment. Agents are trained with the ground-truth hindsight model. The COCOA agent learns slowly compared to the modified approach (R-HCA) we introduce in Section 4.

Evaluation We set the horizon $H = 50$ and the discount factor $\gamma = 1$ for our experiment. We consider $U_k = R_k$, as in COCOA-reward (Meulemans et al., 2023). We train an actor network with data collected episodically using updates and the ground-truth hindsight function,

$$h_k^\pi(s,u) = \begin{cases} \mathbb{I}(a_0) & if \quad u = 1/H, s^4 = 0, s^5 = 0, s^3 = 1, s^2 = 0, \\ \mathbb{I}(a_0) & if \quad u = -1/H, s^4 = 0, s^5 = 0, s^3 = 0, s^2 = 1, \\ \mathbb{I}(a_1) & if \quad u = 1/H, s^4 = 0, s^5 = 0, s^3 = 0, s^2 = 1, \\ \mathbb{I}(a_1) & if \quad u = -1/H, s^4 = 0, s^5 = 0, s^3 = 1, s^2 = 0, \\ \pi(s) & otherwise. \end{cases}$$

We evaluate agents by considering the probability that they take the optimal action in the two possible starting states. Figure 2 shows that even when using the ground truth hindsight model, COCOA provides relatively poor updates for learning a policy.

The failure of COCOA can be attributed to the first term in Equation 4, $\sum_{t\geq 0} \nabla_\theta \log \pi_\theta(A_t|S_t)R_t$, which can dominate the overall sum. This failure mode can be avoided by using an update similar to the original HCA update, assuming that the agent can access an accurate reward model (see Equation 3). Since the immediate rewards in this environment never depend on the action, the gradient $\sum_{t\geq 0} \nabla_\theta \hat{r}(S_t, a)\pi_\theta(a|S_t)$ will be zero.

Our experiment shows that replacing the reward model's predictions with an update that uses the sampled reward may not be a safe choice in general. In the next section we show that the update can be fixed without using a reward model by using our update, which we call R-HCA.

This issue is less prominent in a very similar key-to-door environment considered in the COCOA paper (Meulemans et al., 2023) because there is no noisy reward at the first step where the agent needs to pick up a key.

4 R-HCA

In this section, we propose a modification to COCOA-reward that resolves the issue arising due to how it handles immediate rewards. We show that it is possible to derive a hindsight estimator that weighs the immediate reward with an appropriate hindsight re-weighting.

Recall the COCOA-reward update,

$$\hat{\nabla} V_\pi(S_0) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(A_t|S_t)R_t + \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a|S_t) \sum_{k\geq 1} \left(\frac{\hat{h}_k^\pi(a|S_t, R_{t+k})}{\pi_\theta(a|S_t)} \right)_{sg} \gamma^k R_{t+k}.$$

As illustrated in Section 3, separating immediate and subsequent rewards in this manner can lead to high variance updates due to the first term even when using predictions from the perfect hindsight model.

To overcome this problem, we can modify the update as follows,

$$\hat{\nabla} V_\pi(S_0) = \sum_{t \geq 0} \sum_{a \in \mathcal{A}} \nabla_\theta \pi(a|S_t) \sum_{k \geq 0} \left(\frac{\hat{h}_k^\pi(a|S_t, R_{t+k})}{\pi_\theta(a|S_t)} \right)_{sg} \gamma^k R_{t+k}. \tag{5}$$

We see that this update includes the first reward with hindsight ratio weighting. Next, we prove that this is indeed a valid policy gradient update. The proof is inspired by the work of Alipov et al. (2021), who show that a similar trick can be applied to *state*-HCA when rewards are a function of solely the next state.

Proposition 1. *Consider an action a and state s_t for which $\pi(a|s_t) > 0$. Let $\mathcal{T}(s_t, \pi)$ be distributions over trajectories starting from state s_t when the agent acts with policy π. Then we have*

$$Q_\pi(s_t, a) = \mathbb{E}_{\tau \sim \mathcal{T}(s_t, \pi)} \left[\sum_{k \geq 0} \gamma^k \frac{h_k^\pi\left(a|s_t, R_{t+k}\right)}{\pi\left(a|s_t\right)} R_{t+k} \right].$$

Proof. We can define the action-value function at s_t, a as

$$Q_\pi(s_t, a) = \sum_{r \in \mathcal{R}} \sum_{k \geq 0} \gamma^k P\left(R_{t+k} = r|s_t, a\right) r,$$

where $P\left(R_{t+k} = r|s_t, a_t\right)$ is the probability of getting reward r k steps in the future under the policy-environment distribution and $\mathcal{R}$ is the set of possible values for the reward.

We can apply the Bayes rule:

$$Q_\pi(s_t, a) = \sum_{r \in \mathcal{R}} \sum_{k \geq 0} \gamma^k \frac{P\left(a|s_t, R_{t+k} = r\right) P\left(R_{t+k} = r|s_t\right)}{P\left(a|s_t\right)} r$$

The first term in the numerator is the probability from the hindsight model (by definition), and the denominator is the action probability under the policy. This can be simplified to

$$Q_\pi(s_t, a) = \mathbb{E}_{\tau \sim \mathcal{T}(s_t, \pi)} \left[\sum_{k \geq 0} \gamma^k \frac{h_k^\pi\left(a|s_t, R_{t+k}\right)}{\pi_\theta\left(a|s_t\right)} R_{t+k} \right].$$

□

We obtain the update in Equation 5 by taking an all-action policy gradient. We call our associated algorithm R-HCA (see Algorithm 1).

5 Using learned hindsight models

We revisit the environment introduced in Section 3. Recall that Figure 2 shows that an agent trained with our R-HCA and the ground-truth hindsight model learns very quickly.

Here, we consider performance when the hindsight model is learned together with the actor.

Actor-Critic details The actor and critic are Multi-Layer Perceptrons with no shared parameters with rectified linear (ReLU) units. The actor and critic are updated at the end of each episode. The hindsight-based approaches, R-HCA and COCOA, do not need a critic while operating in episodic settings. Further details are provided in Appendix A.1.

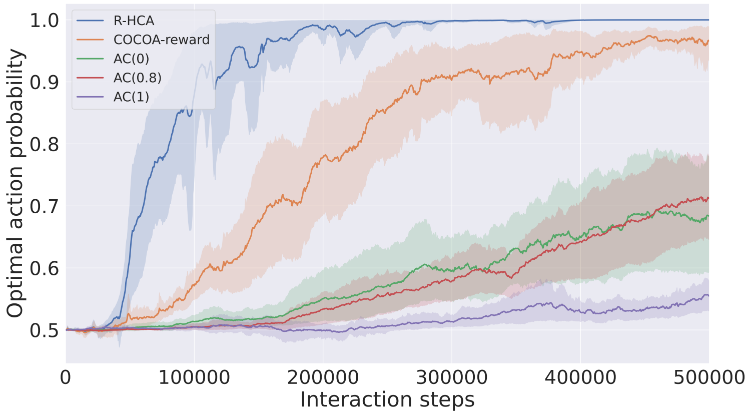

Figure 3: The probability that the policy assigns to the optimal action on the first time step in the umbrella environment. COCOA and R-HCA agents are trained with learned hindsight model. The solid line is the average over 10 seeds, and the shading denotes 95% bootstrapped confidence intervals.

Learning hindsight models The hindsight model takes as input the state, policy logits at state, future reward encoded as a discrete distribution (Imani and White, 2018), and the time elapsed between the state and reward. It predicts a probability distribution over actions as its output. Similar to the actor, the hindsight model is updated on an episodic basis using the data from the latest episode. Further details about the training and architecture are provided in Appendix A.1.

Evaluation We present results for R-HCA, COCOA-reward and A2C(λ), with $\lambda \in \{0, 0.8, 1\}$. We consider different choices of the learning rate and entropy bonus coefficient. For R-HCA and COCOA, we also consider choices for the learning rate of the hindsight model. The considered and selected hyperparameters for all approaches are reported in Appendix A.2. We report the results for the best setting of each approach in in Figure 3.

Figure 3 shows that R-HCA outperforms COCOA-reward when using learned hindsight models. It also shows that both the hindsight-based approaches, R-HCA and COCOA-reward, outperform λ-return-based actor-critic in this problem.

Algorithm 1 R-HCA (Episodic, $\gamma = 1$)

```
for episode=$1, 2 \dots$ do
  $G(i, a) = 0 \quad \forall i, a$
  Collect episode $\tau = (S_0, A_0, R_0, S_1, A_1, R_1, \dots S_H, A_H, R_H, S_{terminal})$ using $\pi_\theta$
  for $i = 0, 1, 2 \dots H$ do
    for $j = i, i + 1 \dots H$ do
      for $a = 1, 2, \dots |\mathcal{A}|$ do
        $G(i, a) = G(i, a) + \frac{\hat{h}^\pi_{j-i}(a|S_i, R_j)}{\pi_\theta(a|S_i)} R_j$
      end for
    end for
  end for
  {Update the actor/policy}
  Take gradient step with loss $L_\pi = -\sum_{i=0}^{H} \sum_{a \in \mathcal{A}} \pi_\theta(a|S_i) G(i, a)$
  {Train the hindsight model}
  Take gradient step with loss $L_h = \sum_{i=0}^{H} \sum_{j=i}^{H} CrossEntropy(\mathbb{I}(A_i), \hat{h}^\pi_{j-i}(.|S_i, R_j))$
end for
```

6 Related Work

Our method extends the Hindsight Credit Assignment (HCA) framework (Harutyunyan et al., 2019). Specifically, we build on the work by Meulemans et al. (2023), who investigate which future features hindsight models should be conditioned on. They propose COCOA, which uses future rewards or features encoding reward-specific information for hindsight-based credit assignment. Compared to COCOA, our R-HCA algorithm improves the assignment of credit associated with immediate rewards.

Several other works have aimed to enhance the original HCA formulation. Some approaches suggest redistributing TD-errors instead of rewards to improve State-HCA (Young, 2019; Alipov et al., 2021), while another work designs a more stable method for estimating the hindsight ratio (Velu et al., 2023). These complementary ideas could potentially be applied with our R-HCA algorithm.

More broadly, HCA can be viewed as a model-based approach to credit assignment. Other model-based methods for credit assignment include differentiating through a learned model (Werbos, 1987; Schmidhuber, 1990; Heess et al., 2015) and using models to chunk or compress trajectories (Ramesh et al., 2024). In contrast to these approaches, which rely on a forward-dynamics model, HCA leverages a multi-step inverse dynamics model to associate rewards with state-action pairs.

HCA also shares similarities with other RL approaches that utilize supervised learning objectives, such as upside-down reinforcement learning (UDRL; Schmidhuber (2019)). UDRL similarly involves learning a hindsight model, typically conditioned on future returns. However, UDRL-like approaches use this model to directly select actions and modify the future information toward more desirable outcomes for policy improvement. In contrast, HCA maintains a separate actor and uses the hindsight model's predictions to increase the estimated advantage.

Beyond HCA and other model-based approaches, several methods aim to improve credit assignment in RL through related mechanisms. These include learning future-conditioned baselines (Mesnard et al., 2021; Nota et al., 2021; Mesnard et al., 2023), retrospectively redistributing returns or identifying critical steps in trajectories (Arjona-Medina et al., 2019; Hung et al., 2019; Sun et al., 2023), and approaches that predict synthetic reward contributions in advance (Raposo et al., 2021). Finally, hindsight re-labeling of goals has proven effective in goal-conditioned RL by enabling more efficient use of collected data (Andrychowicz et al., 2017; Rauber et al., 2019).

7 Conclusion

In this paper, we introduce R-HCA, a credit assignment algorithm that learns a hindsight model to associate future rewards with previous states and actions. Our method addresses an important limitation of the previously proposed COCOA estimator, which exhibits unnecessarily high variance even when using the perfect hindsight model. We resolve this issue by deriving an estimator that incorporates immediate rewards with hindsight re-weighting. Our experimental results on a noisy umbrella problem demonstrate the benefits of our proposed modification.

While hindsight-based approaches show promise, their success has largely been limited to environments with favorable structure, where learning an accurate hindsight model is relatively simple. Future work should focus on evaluating and developing approaches that remain effective even when learning a hindsight model presents greater challenges.

References

Alipov, V., Simmons-Edler, R., Putintsev, N., Kalinin, P., and Vetrov, D. (2021). Towards practical credit assignment for deep reinforcement learning. *arXiv preprint arXiv:2106.04499*.

Amari, S. (1967). A theory of adaptive pattern classifiers. *IEEE Trans. EC*, 16(3):299–307.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. (2017). Hindsight experience replay. *Advances in neural information processing systems*, 30.

Arjona-Medina, J. A., Gillhofer, M., Widrich, M., Unterthiner, T., Brandstetter, J., and Hochreiter, S. (2019). Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR*.

Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515.

Daley, B., Machado, M. C., and White, M. (2024). Demystifying the recency heuristic in temporal-difference learning. *RLJ*, 3:1019–1036.

Glynn, P. W. (1990). Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84.

Harutyunyan, A., Dabney, W., Mesnard, T., Gheshlaghi Azar, M., Piot, B., Heess, N., van Hasselt, H. P., Wayne, G., Singh, S., Precup, D., et al. (2019). Hindsight credit assignment. *Advances in neural information processing systems*, 32:12488–12497.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. (2015). Learning continuous control policies by stochastic value gradients. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Hung, C.-C., Lillicrap, T., Abramson, J., Wu, Y., Mirza, M., Carnevale, F., Ahuja, A., and Wayne, G. (2019). Optimizing agent behavior over long time scales by transporting value. *Nature communications*, 10(1):5223.

Imani, E. and White, M. (2018). Improving regression performance with distributional losses. In *International conference on machine learning*, pages 2157–2166. PMLR.

Mesnard, T., Chen, W., Saade, A., Tang, Y., Rowland, M., Weber, T., Lyle, C., Gruslys, A., Valko, M., Dabney, W., et al. (2023). Quantile credit assignment. In *International Conference on Machine Learning*, pages 24517–24531. PMLR.

Mesnard, T., Weber, T., Viola, F., Thakoor, S., Saade, A., Harutyunyan, A., Dabney, W., Stepleton, T. S., Heess, N., Guez, A., et al. (2021). Counterfactual credit assignment in model-free reinforcement learning. In *International Conference on Machine Learning*, pages 7654–7664. PMLR.

Meulemans, A., Schug, S., Kobayashi, S., Daw, N., and Wayne, G. (2023). Would i have gotten that reward? long-term credit assignment by counterfactual contribution analysis. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PmLR.

Nota, C., Thomas, P., and Da Silva, B. C. (2021). Posterior value functions: Hindsight baselines for policy gradient methods. In *International Conference on Machine Learning*, pages 8238–8247. PMLR.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., et al. (2019). Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*.

Pignatelli, E., Ferret, J., Geist, M., Mesnard, T., van Hasselt, H., and Toni, L. (2024). A survey of temporal credit assignment in deep reinforcement learning. *Trans. Mach. Learn. Res.*, 2024.

Ramesh, A. A., Young, K. J., Kirsch, L., and Schmidhuber, J. (2024). Sequence compression speeds up credit assignment in reinforcement learning. In *Forty-first International Conference on Machine Learning, ICML 2024*.

Raposo, D., Ritter, S., Santoro, A., Wayne, G., Weber, T., Botvinick, M., van Hasselt, H., and Song, F. (2021). Synthetic returns for long-term credit assignment. *arXiv preprint arXiv:2102.12425*.

Rauber, P. E., Ummadisingu, A., Mutz, F., and Schmidhuber, J. (2019). Hindsight policy gradients. In *7th International Conference on Learning Representations, ICLR 2019*.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Schmidhuber, J. (1990). Making the world differentiable: On using fully recurrent self-supervised neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical Report FKI-126-90, `http://people.idsia.ch/~juergen/FKI-126-90_(revised)bw_ocr.pdf`, Tech. Univ. Munich.

Schmidhuber, J. (1992). Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139.

Schmidhuber, J. (2019). Reinforcement learning upside down: Don't predict rewards–just map them to actions. *arXiv preprint arXiv:1912.02875*.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sun, C., Yang, W., Jiralerspong, T., Malenfant, D., Alsbury-Nealy, B., Bengio, Y., and Richards, B. (2023). Contrastive retrospection: honing in on critical steps for rapid learning and generalization in rl. *Advances in Neural Information Processing Systems*, 36:31117–31139.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44.

Sutton, R. S., Barto, A. G., et al. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Velu, A., Vaidyanath, S., and Arumugam, D. (2023). Hindsight-dice: Stable credit assignment for deep reinforcement learning. *arXiv preprint arXiv:2307.11897*.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Werbos, P. J. (1987). Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man, and Cybernetics*, 17.

Williams, R. (1987). A class of gradient-estimation algorithms for reinforcement learning in neural networks. In *Proceedings of the International Conference on Neural Networks*.

Williams, R. J. and Peng, J. (1991). Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268.

Young, K. (2019). Variance reduced advantage estimation with δ-hindsight credit assignment. *arXiv preprint arXiv:1911.08362*.

A Implementation details

A.1 Neural network architectures

Actor-Critic The actor network is composed of 3 hidden layers with 512 hidden units and ReLU non-linearity. The critic network shares no parameters with the actor and is also composed of 3 hidden layers with 512 hidden units and ReLU non-linearity. Note that the hindsight-based approaches (R-HCA and COCOA) do not use a critic in the episodic setting considered here.

Hindsight Model The hindsight model ($\hat{h}_k^\pi(.|s_t, r_{t+k})$) takes as input a state, future reward, policy logits at the state, and the time elapsed between the state and reward. The final output is a logit for each action.

The model architecture consists of a state encoder, a future reward and time-difference encoder, and a gating mechanism to mix the policy log probabilities.

State encoding The state is linearly projected to a vector in $\mathbb{R}^d$, followed by a 3-layer neural network (with ReLU) that maintains the hidden dimension size (d) and by a linear projection to $d \times |\mathcal{A}|$, where $|\mathcal{A}|$ is the number of possible actions. In our experiment, we set $d = 512$.

Reward and time-difference encoding The scalar reward is modified to a 'two-hot' categorical distribution (Imani and White, 2018). We use 51 bins between -0.3 and 0.3 in the considered environment. The two-hot reward is linearly encoded to a d-dimensional vector and additively combined with a learned positional encoding corresponding to the number of timesteps between the state and reward (k in our notation). This is followed by a network similar to the state encoding, a 3-layer neural network (with ReLU) that maintains the hidden dimension size (d) and by a linear projection to $d \times |\mathcal{A}|$.

We obtain the non-gated hindsight model predictions by an element-wise multiplication of the two $d \times |\mathcal{A}|$ feature sets from the state and reward encodings and then sum over the d-dimensions to obtain a prediction vector of size $|\mathcal{A}|$. The non-gated prediction is combined with the policy logits at the state via a gate as we describe below.

Gating We use a gating network to selectively mix the policy logits. With this, the hindsight model only needs to selectively learn the difference between the policy's outputs which helps to handle the problem of changing policies.

The state, reward, and time difference are encoded linearly and pass through a 3-layer network with d hidden units, followed by a final projection to a 2-dimensional output. The gate network outputs two values that lie in $[0, 1]$. The final prediction is obtained as a linear combination of the non-gated prediction and the policy logits.

A.2 Hyperparameter selection

We consider different choices of the learning rate (LR) and entropy bonus coefficient (EB). On-policy actor-critic approaches are commonly trained with an entropy bonus to prevent premature convergence to a deterministic policy (Williams and Peng, 1991; Mnih et al., 2016). We initially considered EB $\in \{0.001, 0.0001, 0.00001\}$ and did not find any approach to be particularly sensitive to the entropy bonus coefficient, so we set it to 0.0001 for all approaches.

The actor and critic networks are trained with the RMSProp optimizer. The considered and selected learning rates for all approaches are reported in Table 1.

For R-HCA and COCOA, we also consider choices for the learning rate of the hindsight model (LRH). The hindsight models are trained with the Adam optimizer. Table 2 presents the considered and selected values.

Table 1: Considered and selected hyperparameters for the actor (and critic in the case of AC).

Algorithm	Candidate LR	Selected LR
AC(0)	$\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$	$5e-5$
AC(0.8)	$\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$	$5e-5$
AC(1)	$\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$	$1e-4$
COCOA-reward	$\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$	$5e-5$
R-HCA	$\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$	$5e-5$

Table 2: Considered and selected hyperparameters for the hindsight model.

Algorithm	Candidate LRH	Selected LRH
COCOA-reward	$\{1e-5, 5e-5, 1e-4, 5e-4\}$	$5e-5$
R-HCA	$\{1e-5, 5e-5, 1e-4, 5e-4\}$	$5e-5$