Would I have gotten that reward? Long-term credit assignment by counterfactual contribution analysis

Alexander Meulemans[*1], **Simon Schug**[*1], **Seijin Kobayashi**[*1]
Nathaniel D Daw[2,3,4], **Gregory Wayne**[2]

[1]Department of Computer Science, ETH Zürich
[2]Google DeepMind
[3]Princeton Neuroscience Institute, Princeton University
[4]Department of Psychology, Princeton University
{ameulema, sschug, seijink}@ethz.ch

Abstract

To make reinforcement learning more sample efficient, we need better credit assignment methods that measure an action's influence on future rewards. Building upon Hindsight Credit Assignment (HCA) [1], we introduce Counterfactual Contribution Analysis (COCOA), a new family of model-based credit assignment algorithms. Our algorithms achieve precise credit assignment by measuring the contribution of actions upon obtaining subsequent rewards, by quantifying a counterfactual query: 'Would the agent still have reached this reward if it had taken another action?'. We show that measuring contributions w.r.t. rewarding *states*, as is done in HCA, results in spurious estimates of contributions, causing HCA to degrade towards the high-variance REINFORCE estimator in many relevant environments. Instead, we measure contributions w.r.t. rewards or learned representations of the rewarding objects, resulting in gradient estimates with lower variance. We run experiments on a suite of problems specifically designed to evaluate long-term credit assignment capabilities. By using dynamic programming, we measure ground-truth policy gradients and show that the improved performance of our new model-based credit assignment methods is due to lower bias and variance compared to HCA and common baselines. Our results demonstrate how modeling action contributions towards rewarding outcomes can be leveraged for credit assignment, opening a new path towards sample-efficient reinforcement learning.[2]

1 Introduction

Reinforcement learning (RL) faces two central challenges: exploration and credit assignment [2]. We need to explore to discover rewards and we need to reinforce the actions that are instrumental for obtaining these rewards. Here, we focus on the credit assignment problem and the intimately linked problem of estimating policy gradients. For long time horizons, obtaining the latter is notoriously difficult as it requires measuring how each action influences expected subsequent rewards. As the number of possible trajectories grows exponentially with time, future rewards come with a considerable variance stemming from stochasticity in the environment itself and from the stochasticity of interdependent future actions leading to vastly different returns [3–5].

Monte Carlo estimators such as REINFORCE [6] therefore suffer from high variance, even after variance reduction techniques like subtracting a baseline [6–9]. Similarly, in Temporal Difference methods such as Q-learning, this high variance in future rewards results in a high bias in the value estimates, requiring exponentially many updates to correct for it [5]. Thus, a common technique to

[*]Equal contribution; ordering determined by coin flip.
[2]Code available at `https://github.com/seijin-kobayashi/cocoa`

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

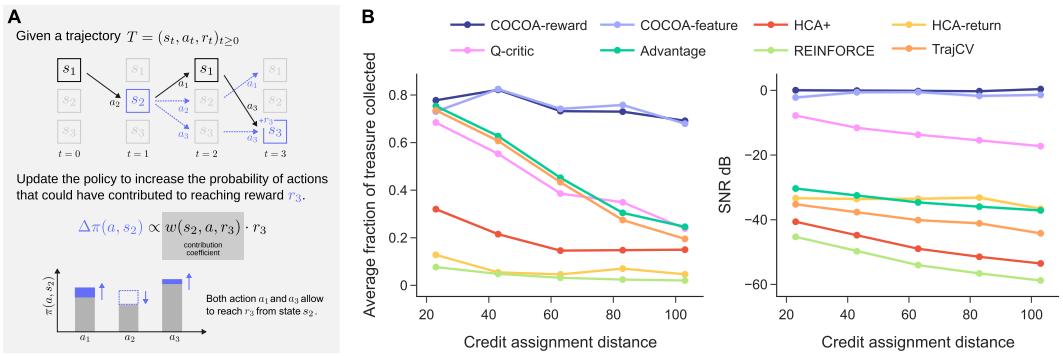

Figure 1: **Counterfactual Contribution Analysis enables long-term credit assignment.** (A) Given a sample trajectory that eventually results in a rewarding outcome, we estimate the policy gradient by considering the contribution of actions along the trajectory towards arriving at a rewarding outcome. In this example, we measure how much more likely the rewarding outcome with reward r_3 is when following action a_1 versus the counterfactual actions a_2 and a_3 in state s_2. This is quantified through the contribution coefficient $w(s_2, a_1, r_3)$ which is used to update all possible action probabilities of the policy $\pi(a \mid s_2)$. (B) In the linear key-to-door environment increasing the distance between picking up the key and opening the door that leads to reward necessitates credit assignment over increasing time spans. COCOA consistently achieves good performance (left) compared to HCA and baselines which deteriorate when increasing the distance between an action and the resulting rewarding outcome. This is reflected in a higher signal-to-noise ratio of the policy gradient estimator of COCOA compared to baselines (right).

reduce variance and bias is to discount rewards that are far away in time resulting in a biased estimator which ignores long-term dependencies [10–13]. Impressive results have nevertheless been achieved in complex environments [14–16] at the cost of requiring billions of environment interactions, making these approaches sample inefficient.

Especially in settings where obtaining such large quantities of data is costly or simply not possible, model-based RL that aims to simulate the dynamics of the environment is a promising alternative. While learning such a world model is a difficult problem by itself, when successful it can be used to generate a large quantity of synthetic environment interactions. Typically, this synthetic data is combined with model-free methods to improve the action policy [17–19]. A notable exception to simply using world models to generate more data are the Stochastic Value Gradient method [20] and the closely related Dreamer algorithms [21–23]. These methods perform credit assignment by backpropagating policy gradients through the world model. Crucially, this approach only works for environments with a continuous state-action space, as otherwise sensitivities of the value with respect to past actions are undefined [20, 22, 24]. Intuitively, we cannot compute sensitivities of discrete choices such as a yes / no decision as the agent cannot decide 'yes' a little bit more or less.

Building upon Hindsight Credit Assignment (HCA) [1], we develop Counterfactual Contribution Analysis (COCOA), a family of algorithms that use models for credit assignment compatible with discrete actions. We measure the *contribution* of an action upon subsequent rewards by asking a counterfactual question: 'would the agent still have achieved the rewarding outcome, if it had taken another action?' (c.f. Fig. 1A). We show that measuring contributions towards achieving a future *state*, as is proposed in HCA, leads to spurious contributions that do not reflect a contribution towards a reward. This causes HCA to degrade towards the high-variance REINFORCE method in most environments. Instead, we propose to measure contributions directly on rewarding outcomes and we develop various new ways of learning these contributions from observations. The resulting algorithm differs from value-based methods in that it measures the contribution of an action to individual rewards, instead of estimating the full expected sum of rewards. This crucial difference allows our contribution analysis to disentangle different tasks and ignore uncontrollable environment influences, leading to a gradient estimator capable of long-term credit assignment (c.f. Fig. 1B). We introduce a new method for analyzing policy gradient estimators which uses dynamic programming to allow comparing to ground-truth policy gradients. We leverage this to perform a detailed bias-variance analysis of all proposed methods and baselines showing that our new model-based credit assignment algorithms achieve low variance and bias, translating into improved performance (c.f. Fig. 1C).

2 Background and notation

We consider an undiscounted Markov decision process (MDP) defined as the tuple $(\mathcal{S}, \mathcal{A}, p, p_r)$, with $\mathcal{S}$ the state space, $\mathcal{A}$ the action space, $p(S_{t+1} \mid S_t, A_t)$ the state-transition distribution and $p_r(R \mid S, A)$ the reward distribution with bounded reward values r. We use capital letters for random variables and lowercase letters for the values they take. The policy $\pi(A \mid S)$, parameterized by θ, denotes the probability of taking action A at state S. We consider an undiscounted infinite-horizon setting with a zero-reward absorbing state s_∞ that the agent eventually reaches: $\lim_{t\to\infty} p(S_t = s_\infty) = 1$. Both the discounted and episodic RL settings are special cases of this setting. (c.f. App. B), and hence all theoretical results proposed in this work can be readily applied to both (c.f. App J).

We use $\mathcal{T}(s, \pi)$ and $\mathcal{T}(s, a, \pi)$ as the distribution over trajectories $T = (S_t, A_t, R_t)_{t\geq 0}$ starting from $S_0 = s$ and $(S_0, A_0) = (s, a)$ respectively, and define the return $Z_t = \sum_{t=0}^{\infty} R_t$. The value function $V^\pi(s) = \mathbb{E}_{T\sim\mathcal{T}(s,\pi)}[Z_t]$ and action value function $Q^\pi(s, a) = \mathbb{E}_{T\sim\mathcal{T}(s,a,\pi)}[Z_t]$ are the expected return when starting from state s, or state s and action a respectively. Note that these infinite sums have finite values due to the absorbing zero-reward state (c.f. App. B).

The objective of reinforcement learning is to maximize the expected return $V^\pi(s_0)$, where we assume the agent starts from a fixed state s_0. Policy gradient algorithms optimize $V^\pi(s_0)$ by repeatedly estimating its gradient $\nabla_\theta V(s_0)$ w.r.t. the policy parameters. REINFORCE [6] (c.f. Tab. 1) is the canonical policy gradient estimator, however, it has a high variance resulting in poor parameter updates. Common techniques to reduce the variance are (i) subtracting a baseline, typically a value estimate, from the sum of future rewards [2, 25] (c.f. 'Advantage' in Tab. 1); (ii) replacing the sum of future rewards with a learned action value function Q [2, 3, 25, 26] (c.f. 'Q-critic' in Tab. 1); and (iii) using temporal discounting. Note that instead of using a discounted formulation of MDPs, we treat the discount factor as a variance reduction technique in the undiscounted problem [10, 11, 13] as this more accurately reflects its practical use [4, 27]. Rearranging the summations of REINFORCE with discounting lets us interpret temporal discounting as a credit assignment heuristic, where for each reward, past actions are reinforced proportional to their proximity in time.

$$\hat{\nabla}_\theta^{\text{REINFORCE},\gamma} V^\pi(s_0) = \sum\nolimits_{t\geq 0} R_t \sum\nolimits_{k\leq t} \gamma^{t-k} \nabla_\theta \log \pi(A_k \mid S_k), \quad \gamma \in [0, 1]. \tag{1}$$

Crucially, long-term dependencies between actions and rewards are exponentially suppressed, thereby reducing variance at the cost of disabling long-term credit assignment [4, 28]. The aim of this work is to replace the heuristic of time discounting by principled *contribution coefficients* quantifying how much an action contributed towards achieving a reward, and thereby introducing new policy gradient estimators with reduced variance, without jeopardizing long-term credit assignment.

HCA [1] makes an important step in this direction by introducing a new gradient estimator:

$$\hat{\nabla}_\theta^{\text{HCA}} V^\pi = \sum_{t\geq 0} \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) \Big(r(S_t, a) + \sum_{k\geq 1} \frac{p^\pi(A_t = a \mid S_t = s, S' = S_{t+k})}{\pi(a \mid S_t)} R_{t+k} \Big) \tag{2}$$

with $r(s, a)$ a reward model, and the *hindsight* ratio $\frac{p^\pi(a|S_t=s,S'=S_{t+k})}{\pi(a|S_t)}$ measuring how important action a was to reach the state S' at some point in the future. Although the hindsight ratio delivers precise credit assignment w.r.t. reaching future states, it has a failure mode of practical importance, creating the need for an updated theory of model-based credit assignment which we will detail in the next section.

3 Counterfactual Contribution Analysis

To formalize the 'contribution' of an action upon subsequent rewards, we generalize the theory of HCA [1] to measure contributions on *rewarding outcomes* instead of states. We introduce unbiased policy gradient estimators that use these contribution measures, and show that HCA suffers from high variance, making the generalization towards rewarding outcomes crucial for obtaining low-variance estimators. Finally, we show how we can estimate contributions using observational data.

3.1 Counterfactual contribution coefficients

To assess the contribution of actions towards rewarding outcomes, we propose to use counterfactual reasoning: 'how does taking action a influence the probability of obtaining a rewarding outcome, compared to taking alternative actions a'?'.

Table 1: Comparison of policy gradient estimators.

Method	**Policy gradient estimator** ($\hat{\nabla}_\theta V^\pi(s_0)$)
REINFORCE	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) \sum_{k\geq 0} R_{t+k}$
Advantage	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) \Big(\sum_{k\geq 0} R_{t+k} - V(S_t)\Big)$
Q-critic	$\sum_{t\geq 0} \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) Q(S_t, a)$
HCA-Return	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) \Big(1 - \frac{\pi(A_t\mid S_t)}{p^\pi(A_t\mid S_t, Z_t)}\Big) Z_t$
TrajCV	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) \big(Z_t - Q(A_t, S_t) - \sum_{t'>t} \big(Q(S_{t'}, A_{t'}) - V(S_{t'})\big) + \ldots$ $\ldots \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) Q(S_t, a)$
COCOA	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) R_t + \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) \sum_{k\geq 1} w(S_t, a, U_{t+k}) R_{t+k}$
HCA+	$\sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) R_t + \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) \sum_{k\geq 1} w(S_t, a, S_{t+k}) R_{t+k}$

Definition 1 (Rewarding outcome)**.** A rewarding outcome $U' \sim p(U' \mid s', a', r')$ is a probabilistic encoding of the state-action-reward triplet.

If action a contributed towards the reward, the probability of obtaining the rewarding outcome following action a should be higher compared to taking alternative actions. We quantify the contribution of action a taken in state s upon rewarding outcome u' as

$$w(s, a, u') = \frac{\sum_{k\geq 1} p^\pi(U_{t+k} = u' \mid S_t = s, A_t = a)}{\sum_{k\geq 1} p^\pi(U_{t+k} = u' \mid S_t = s)} - 1 = \frac{p^\pi(A_t = a \mid S_t = s, U' = u')}{\pi(a \mid s)} - 1 \qquad (3)$$

From a given state, we compare the probability of reaching the *rewarding outcome* u' at any subsequent point in time, given we take action a versus taking counterfactual actions according to the policy π, as $p^\pi(U_{t+k} = u' \mid S_t = s) = \sum_{a'} \pi(a' \mid s) p^\pi(U_{t+k} = u' \mid S_t = s, A_t = a')$. Subtracting this ratio by one results in an intuitive interpretation of the *contribution coefficient* $w(s, a, u')$: if the coefficient is positive/negative, performing action a results in a higher/lower probability of obtaining rewarding outcomes u', compared to following the policy π. Using Bayes' rule, we can convert the counterfactual formulation of the contribution coefficients into an equivalent *hindsight* formulation (right-hand side of Eq. 3), where the hindsight distribution $p^\pi(A_t = a \mid S_t = s, U' = u')$ reflects the probability of taking action a in state s, given that we encounter the rewarding outcome u' at *some future point in time*. We refer the reader to App. C for a full derivation.

Choice of rewarding outcome. For $u' = s'$, we recover state-based HCA [1].[3] In the following, we show that a better choice is to use $u' = r'$, or an encoding $p(u' \mid s', a')$ of the underlying object that causes the reward. Both options lead to gradient estimators with lower variance (c.f. Section 3.3), while using the latter becomes crucial when different underlying rewarding objects have the same scalar reward (c.f. Section 4).

3.2 Policy gradient estimators

We now show how the contribution coefficients can be used to learn a policy. Building upon HCA [1], we propose the Counterfactual Contribution Analysis (COCOA) policy gradient estimator

$$\hat{\nabla}^U_\theta V^\pi(s_0) = \sum_{t\geq 0} \nabla_\theta \log \pi(A_t \mid S_t) R_t + \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S_t) \sum_{k\geq 1} w(S_t, a, U_{t+k}) R_{t+k}. \qquad (4)$$

When comparing to the discounted policy gradient of Eq. 1, we see that the temporal discount factors are substituted by the contribution coefficients, replacing the time heuristic with fine-grained credit assignment. Importantly, the contribution coefficients enable us to evaluate all counterfactual actions instead of only the observed ones, further increasing the quality of the gradient estimator (c.f. Fig. 1A). The contribution coefficients allow for various different gradient estimators (c.f. App. C). For example, independent action samples can replace the sum over all actions, making it applicable to large action spaces. Here, we use the gradient estimator of Eq. 4, as our experiments consist of small action spaces where enumerating all actions is feasible. When $U = S$, the above estimator is almost equivalent to

[3]Note that Harutyunyan et al. [1] also introduce an estimator based on the return. Here, we focus on the state-based variant as only this variant uses contribution coefficients to compute the policy gradient (Eq. 4). The return-based variant instead uses the hindsight distribution as an action-dependent baseline as shown in Tab. 1. Importantly, the return-based estimator is biased in many relevant environments (c.f. Appendix L).

the state-based HCA estimator of Eq. 2, except that it does not need a learned reward model $r(s, a)$. We use the notation HCA+ to refer to this simplified version of the HCA estimator. Theorem 1 below shows that the COCOA gradient estimator is unbiased, as long as the encoding U is fully predictive of the reward, thereby generalizing the results of Harutyunyan et al. [1] to arbitrary rewarding outcome encodings.

Definition 2 (Fully predictive)**.** A rewarding outcome U is fully predictive of the reward R, if the following conditional independence condition holds for all $k \geq 0$: $p^\pi(R_k = r \mid S_0 = s, A_0 = a, U_k = u) = p^\pi(R = r \mid U = u)$, where the right-hand side does not depend on the time k.

Theorem 1. *Assuming that U is fully predictive of the reward (c.f. Definition 2), the COCOA policy gradient estimator $\hat{\nabla}^U_\theta V^\pi(s_0)$ is unbiased, when using the ground-truth contribution coefficients of Eq. 3, that is*

$$\nabla_\theta V^\pi(s_0) = \mathbb{E}_{T \sim \mathcal{T}(s_0, \pi)} \hat{\nabla}^U_\theta V^\pi(s_0).$$

3.3 Optimal rewarding outcome encoding for low-variance gradient estimators

Theorem 1 shows that the COCOA gradient estimators are unbiased for all rewarding outcome encodings U that are fully predictive of the reward. The difference between specific rewarding outcome encodings manifests itself in the variance of the resulting gradient estimator. Proposition 2 shows that for $U' = S'$ as chosen by HCA there are many cases where the variance of the resulting policy gradient estimator degrades to the high-variance REINFORCE estimator [6]:

Proposition 2. *In environments where each action sequence leads deterministically to a different state, we have that the HCA+ estimator is equal to the REINFORCE estimator (c.f. Tab. 1).*

In other words, when all previous actions can be perfectly decoded from a given state, they trivially all contribute to reaching this state. The proof of Propostion 2 follows immediately from observing that $p^\pi(a \mid s, s') = 1$ for actions a along the observed trajectory, and zero otherwise. Substituting this expression into the contribution analysis gradient estimator (4) recovers REINFORCE. A more general issue underlies this special case: State representations need to contain detailed features to allow for a capable policy but the same level of detail is detrimental when assigning credit to actions for reaching a particular state since at some resolution almost every action will lead to a slightly different outcome. Measuring the contribution towards reaching a specific state ignores that the same rewarding outcome could be reached in slightly different states, hence overvaluing the importance of previous actions and resulting in *spurious contributions*. Many commonly used environments, such as pixel-based environments, continuous environments, and partially observable MDPs exhibit this property to a large extent due to their fine-grained state representations (c.f. App. G). Hence, our generalization of HCA to rewarding outcomes is a crucial step towards obtaining practical low-variance gradient estimators with model-based credit assignment.

Using rewards as rewarding outcomes yields lowest-variance estimators. The following Theorem 3 shows in a simplified setting that (i) the variance of the REINFORCE estimator is an upper bound on the variance of the COCOA estimator, and (ii) the variance of the COCOA estimator is smaller for rewarding outcome encodings U that contain less information about prior actions. We formalize this with the conditional independence relation of Definition 2 by replacing R with U': encoding U contains less or equal information than encoding U', if U' is fully predictive of U. Combined with Theorem 1 that states that an encoding U needs to be fully predictive of the reward R, we have that taking the reward R as our rewarding outcome encoding U results in the gradient estimator with the lowest variance of the COCOA family.

Theorem 3. *Consider an MDP where only the states at a single (final) time step contain a reward, and where we optimize the policy only at a single (initial) time step. Furthermore, consider two rewarding outcome encodings U and U', where S is fully predictive of U', U' fully predictive of U, and U fully predictive of R. Then, the following relation holds between the policy gradient estimators:*

$$\mathbb{V}[\hat{\nabla}^R_\theta V^\pi(s_0)] \preccurlyeq \mathbb{V}[\hat{\nabla}^U_\theta V^\pi(s_0)] \preccurlyeq \mathbb{V}[\hat{\nabla}^{U'}_\theta V^\pi(s_0)] \preccurlyeq \mathbb{V}[\hat{\nabla}^S_\theta V^\pi(s_0)] \preccurlyeq \mathbb{V}[\hat{\nabla}^{REINF}_\theta V^\pi(s_0)]$$

with $\hat{\nabla}^X_\theta V^\pi(s_0)$ the COCOA estimator (4) using $U = X$, $\mathbb{V}[Y]$ the covariance matrix of Y and $A \preccurlyeq B$ indicating that $B - A$ is positive semi-definite.

As Theorem 3 considers a simplified setting, we verify empirically whether the same arguments hold more generally. We construct a tree environment where we control the amount of information a

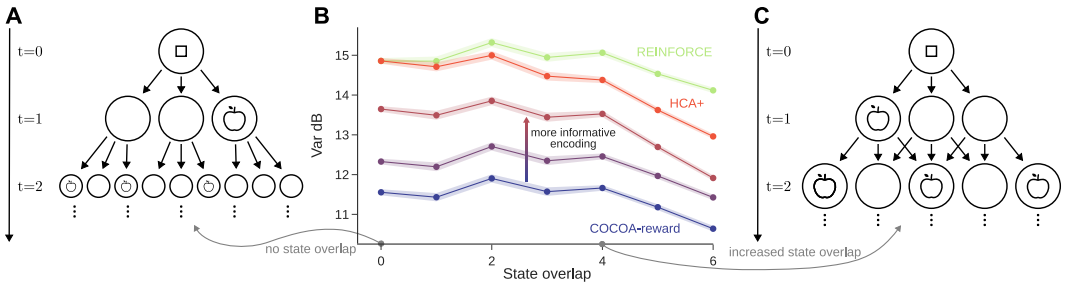

Figure 2: **HCA suffers from spurious contributions which can be alleviated by using less informative rewarding outcome encodings.** (A) and (C): Schematic of the tree environment where we parametrically adjust the amount of overlap between states by varying the amount of shared children of two neighboring nodes. We can decrease the information content of the rewarding outcome encoding $u = f(s, a)$ by grouping state-action pairs that share the same reward value. (B) Normalized variance in dB using ground-truth coefficients and a random uniform policy (shaded region represents standard error over 10 random environments) comparing REINFORCE, HCA, COCOCA-reward and various degrees of intermediate grouping.

state contains about the previous actions by varying the overlap of the children of two neighbouring nodes (c.f. Fig 2), and assign a fixed random reward to each state-action pair. We compute the ground-truth contribution coefficients by leveraging dynamic programming (c.f. Section 4). Fig. 2B shows that the variance of HCA is as high as REINFORCE for zero state overlap, but improves when more states overlap, consistent with Proposition 2 and Theorem 3. To investigate the influence of the information content of U on the variance, we consider rewarding outcome encodings U with increasing information content, which we quantify with how many different values of u belong to the same reward r. Fig. 2B shows that by increasing the information content of U, we interpolate between the variance of COCOA with $u = r$ and HCA+, consistent with Theorem 3.

Why do rewarding outcome encodings that contain more information than the reward lead to higher variance? To provide a better intuition on this question we use the following theorem:

Theorem 4. *The policy gradient on the expected number of occurrences* $O^\pi(u', s) = \sum_{k\geq 1} p^\pi(U_k = u' \mid S_0 = s)$ *is proportional to*

$$\nabla_\theta O^\pi(u', s) \propto \mathbb{E}_{S''\sim\mathcal{T}(\pi,s)} \sum\nolimits_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid S'') w(S'', a, u') O^\pi(u', S'')$$

Recall that the COCOA gradient estimator consists of individual terms that credit past actions a at state s for a current reward r encountered in u according to $\sum_{a\in\mathcal{A}} \nabla_\theta \pi(a \mid s) w(s, a, u') r'$ (c.f. Eq. 4). Theorem 4 indicates that each such term aims to increase the average number of times we encounter u' in a trajectory starting from s, proportional to the corresponding reward r'. If $U' = R'$, this update will correctly make all underlying states with the same reward r' more likely while decreasing the likeliness of all states for which $u' \neq r'$. Now consider the case where our rewarding outcome encoding contains a bit more information, i.e. $U' = f(R', \Delta S')$ where $\Delta S'$ contains a little bit of information about the state. As a result the update will distinguish some states even if they yield the same reward and increase the number of occurrences only of states containing the encountered $\Delta S'$ while decreasing the number of occurrences for unseen ones. As in a single trajectory, we do not visit each possible $\Delta S'$, this adds variance. The less information an encoding U has, the more underlying states it groups together, and hence the less rewarding outcomes are 'forgotten' in the gradient estimator, leading to lower variance.

3.4 Learning the contribution coefficients

In practice, we do not have access to the ground-truth contribution coefficients, but need to learn them from observations. Following Harutyunyan et al. [1], we can approximate the hindsight distribution $p^\pi(A_t = a \mid S_t = s, U' = u')$, now conditioned on rewarding outcome encodings instead of states, by training a model $h(a \mid s, u')$ on the supervised discriminative task of classifying the observed action a_t given the current state s_t and some future rewarding outcome u'. Note that if the model h does not approximate the hindsight distribution perfectly, the COCOA gradient estimator (4) can be biased (c.f. Section 4). A central difficulty in approximating the hindsight distribution is that it is policy dependent, and hence changes during training. Proposition 5 shows that we can provide the policy logits as an extra input to the hindsight network without altering the learned hindsight

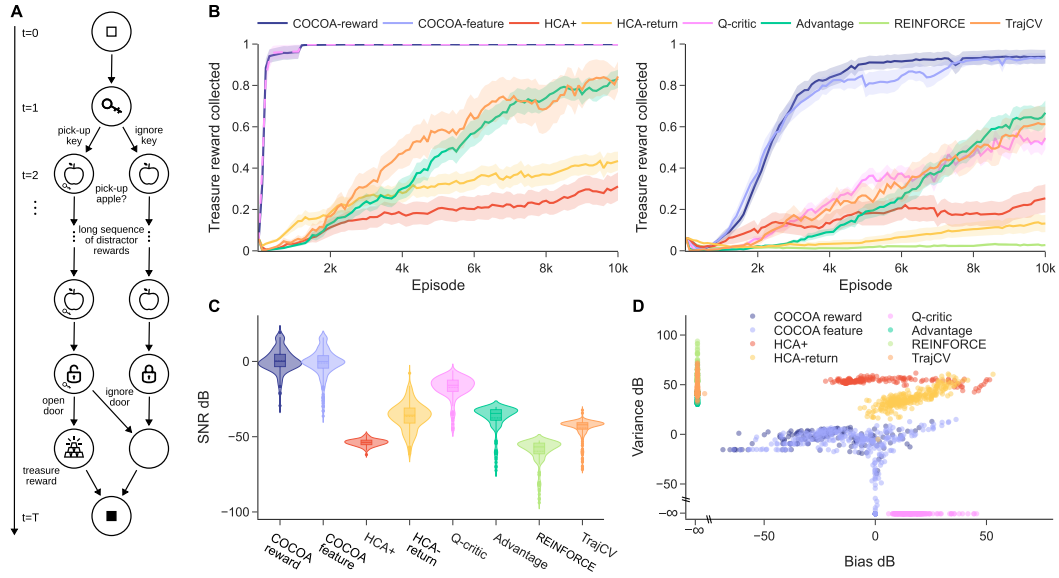

Figure 3: **COCOA enhances policy gradient estimates and sample efficiency whereas HCA fails to improve over baselines.** (A) Schematic representation of the linear key-to-door environment. (B) Performance of COCOA and baselines on the main task of picking up the treasure, measured as the average fraction of treasure rewards collected. (Left) ground-truth policy gradient estimators computed using dynamic programming, (right) learning the contribution coefficients or (action-)value function using neural networks. Shaded regions are the standard error (30 seeds). (C) Violin plot of the signal-to-noise ratio (SNR) in Decibels for the various policy gradient estimators with learned coefficients and (action-)value functions, computed on the same trajectories of a shared base policy. (D) Comparison of the bias-variance trade-off incurred by different policy gradient estimators, computed as in (C), normalized by the ground-truth policy gradient norm (scatter plot showing 30 seeds per method).

distribution, to make the parameters of the hindsight network less policy-dependent. This observation justifies and generalizes the strategy of adding the policy logits to the hindsight model output, as proposed by Alipov et al. [29].

Proposition 5. $p^\pi(a \mid s, u', l) = p^\pi(a \mid s, u')$, *with* l *a deterministic function of* s*, representing the sufficient statistics of* $\pi(a \mid s)$.

As an alternative to learning the hindsight distribution, we can directly estimate the probability ratio $p^\pi(A_t = a \mid S_t = s, U' = u')/\pi(a \mid s)$ using a contrastive loss (c.f. App. D). Yet another path builds on the observation that the sums $\sum_{k\geq 1} p^\pi(U_{t+k} = u' \mid s, a)$ are akin to Successor Representations and can be learned via temporal difference updates [30, 31] (c.f. App. D). We experimented both with the hindsight classification and the contrastive loss and found the former to work best in our experiments. We leverage the Successor Representation to obtain ground truth contribution coefficients via dynamic programming for the purpose of analyzing our algorithms.

4 Experimental analysis

To systematically investigate long-term credit assignment performance of COCOA compared to standard baselines, we design an environment which pinpoints the core credit assignment problem and leverage dynamic programming to compute ground-truth policy gradients, contribution coefficients, and value functions (c.f. App E.2). This enables us to perform detailed bias-variance analyses and to disentangle the theoretical optimal performance of the various gradient estimators from the approximation quality of learned contribution coefficients and (action-)value functions.

We consider the *linear key-to-door* environment (c.f. Fig. 3A), a simplification of the key-to-door environment [3, 4, 32] to a one-dimensional track. Here, the agent needs to pick up a key in the first time step, after which it engages in a distractor task of picking up apples with varying reward values. Finally, it can open a door with the key and collect a treasure. This setting allows us to parametrically increase the difficulty of long-term credit assignment by increasing the distance between the key and door, making it harder to pick up the learning signal of the treasure reward

among a growing number of varying distractor rewards [4]. We use the signal-to-noise ratio, SNR$=\|\nabla_\theta V^\pi\|^2/\mathbb{E}[\|\hat{\nabla}_\theta V^\pi - \nabla_\theta V^\pi\|^2]$, to quantify the quality of the different policy gradient estimators; a higher SNR indicates that we need fewer trajectories to estimate accurate policy gradients [33].

Previously, we showed that taking the reward as rewarding outcome encoding results in the lowest-variance policy gradients when using ground-truth contribution coefficients. In this section, we will argue that when *learning* the contribution coefficients, it is beneficial to use an encoding u of the underlying *rewarding object* since this allows to distinguish different rewarding objects when they have the same scalar reward value and allows for quick adaptation when the reward function but not the environment dynamics changes.

We study two variants of COCOA, COCOA-reward which uses the reward identity for U, and COCOA-feature which acquires features of rewarding objects by learning a reward model $r(s, a)$ and taking the penultimate network layer as U. We learn the contribution coefficients by approximating the hindsight distribution with a neural network classifier $h(a \mid s, u', l)$ that takes as input the current state s, resulting policy logits l, and rewarding outcome u', and predicts the current action a (c.f. App. E for all experimental details). As HCA+ (c.f. Tab. 1) performs equally or better compared to HCA [1] in our experiments (c.f. App. F), we compare to HCA+ and several other baselines: (i) three classical policy gradient estimators, REINFORCE, Advantage and Q-critic, (ii) TrajCV [34], a state-of-the-art control variate method that uses hindsight information in its baseline, and (iii) HCA-return [1], a different HCA variant that uses the hindsight distribution conditioned on the return as an action-dependent baseline (c.f. Tab. 1).

4.1 COCOA improves sample-efficiency due to favorable bias-variance trade-off.

To investigate the quality of the policy gradient estimators of COCOA, we consider the linear key-to-door environment with a distance of 100 between key and door. Our dynamic programming setup allows us to disentangle the performance of the estimators independent of the approximation quality of learned models by using ground truth contribution coefficients and (action-)value functions. The left panel of figure 3B reveals that in this ground truth setting, COCOA-reward almost immediately solves the task performing as well as the theoretically optimal Q-critic with a perfect action-value function. This is in contrast to HCA and HCA-return which perform barely better than REINFORCE, all failing to learn to consistently pick up the key in the given number of episodes. This result translates to the setting of learning the underlying models using neural networks. COCOA-reward and -feature outperform competing policy gradient estimators in terms of sample efficiency while HCA only improves over REINFORCE. Notably, having to learn the full action-value function leads to a less sample-efficient policy gradient estimator for the Q-critic.

In Figure 3C and D we leverage dynamic programming to compare to the ground truth policy gradient. This analysis reveals that improved performance of COCOA is reflected in a higher SNR compared to other estimators due to its favorable bias-variance trade-off. Fig 12 in App. F indicates that COCOA maintains a superior SNR, even when using significantly biased contribution coefficients. As predicted by our theory in Section 3.3, HCA significantly underperforms compared to baselines due to its high variance caused by spurious contributions. In particular, the Markov state representation of the linear key-to-door environment contains the information of whether the key has been picked up. As a result, HCA always credits picking up the key or not, even for distractor rewards. These spurious contributions bury the useful learning signal of the treasure reward in noisy distractor rewards. HCA-return performs poorly as it is a biased gradient estimator, even when using the ground-truth hindsight distribution (c.f. Appendix F and L). Interestingly, the variance of COCOA is significantly lower compared to a state-of-the-art control variate method, TrajCV, pointing to a potential benefit of the multiplicative interactions between contribution coefficients and rewards in the COCOA estimators, compared to the additive interaction of the control variates: the value functions used in TrajCV need to approximate the full average returns, whereas COCOA can ignore rewards from the distractor subtask, by multiplying them with a contribution coefficient of zero.

4.2 COCOA enables long-term credit assignment by disentangling rewarding outcomes.

The linear key-to-door environment allows us to parametrically increase the difficulty of the long-term credit assignment problem by increasing the distance between the key and door and thereby increasing the variance due to the distractor task [4]. Figure 1B reveals that as this distance increases, performance measured as the average fraction of treasure collected over a fixed number of episodes

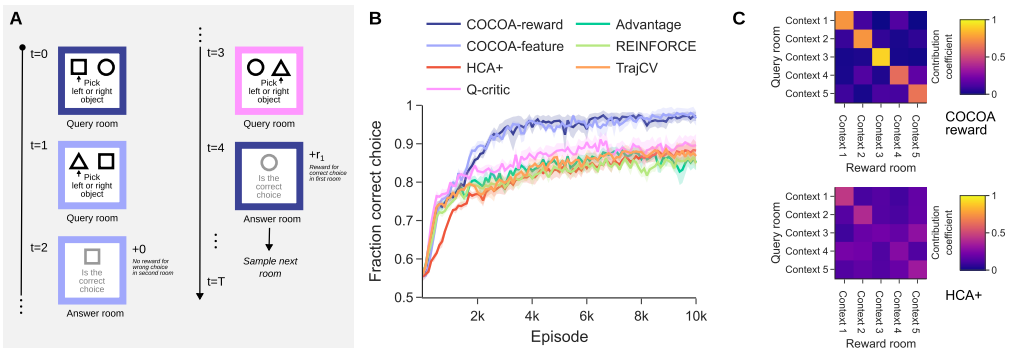

Figure 4: **COCOA improves performance by disentangling subtasks.** (A) Schematic representation of the task interleaving environment where colored borders indicate the context of a room. (B) Performance of COCOA and baselines with learned contribution coefficients or value functions, measured as the fraction of correct choices. (C) Visualization of the contribution coefficient magnitudes of each query room on reward rooms for COCOA (top) and HCA (bottom), c.f. App. E.7.

drops for all baselines including HCA but remains relatively stable for the COCOA estimators. Hung et al. [4] showed that the SNR of REINFORCE decreases inversely proportional to growing distance between key and door. Figure 1B shows that HCA and all baselines follow this trend, whereas the COCOA estimators maintain a robust SNR.

We can explain the qualitative difference between COCOA and other methods by observing that the key-to-door task consists of two distinct subtasks: picking up the key to get the treasure, and collecting apples. COCOA can quickly learn that actions relevant for one task, do not influence rewarding outcomes in the other task, and hence output a contribution coefficient equal to zero for those combinations. Value functions in contrast estimate the expected sum of future rewards, thereby mixing the rewards of both tasks. When increasing the variance of the return in the distractor task by increasing the number of stochastic distractor rewards, estimating the value functions becomes harder, whereas estimating the contribution coefficients between state-action pairs and distinct rewarding objects remains of equal difficulty, showcasing the power of disentangling rewarding outcomes.

To further showcase the power of disentangling subtasks, we consider a simplified version of the *task interleaving* environment of Mesnard et al. [3] (c.f. Fig. 4A, App. E.7). Here, the agent is faced with a sequence of contextual bandit tasks, where the reward for a correct decision is given at a later point in time, together with an observation of the relevant context. The main credit assignment difficulty is to relate the reward and contextual observation to the correct previous contextual bandit task. Note that the variance in the return is now caused by the stochastic policy, imperfectly solving future tasks, and by stochastic state transitions, in contrast to the linear key-to-door environment where the variance is caused by stochastic distractor rewards. Figure 4B shows that our COCOA algorithms outperform all baselines. The learned contribution coefficients of COCOA reward accurately capture that actions in one context only contribute to rewards in the same context as opposed to HCA that fails to disentangle the contributions (c.f. Figure 4C).

4.3 Learned credit assignment features allow for disentangling aliased rewards

For COCOA-reward we use the scalar reward value to identify rewarding outcomes in the hindsight distribution, i.e. $U = R$. In cases where multiple rewarding outcomes yield an identical scalar reward value, the hindsight distribution cannot distinguish between them and has to estimate a common hindsight

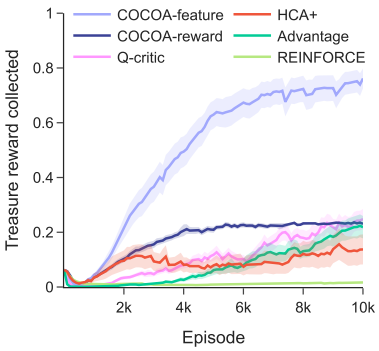

Figure 5: **COCOA-features is robust to reward aliasing**. On a version of the linear key-to-door environment where one of the distractor reward values has the same magnitude as the treasure reward, COCOA-reward can no longer distinguish between the distractor and treasure reward and as a result its performance decreases. COCOA-feature is robust to this manipulation since it relies on learned features to distinguish rewarding objects.

probability, making it impossible to disentangle the tasks and hence rendering learning of the contribution coefficients potentially more difficult. In contrast, COCOA-feature learns hindsight features of rewarding objects that are predictive of the reward. Even when multiple rewarding objects lead to an identical scalar reward value their corresponding features are likely different, allowing COCOA-feature to disentangle the rewarding outcomes.

In Fig. 5, we test this *reward aliasing* setting experimentally and slightly modify the linear key-to-door environment by giving the treasure reward the same value as one of the two possible values of the stochastic distractor rewards. As expected, COCOA-feature is robust to reward aliasing, continuing to perform well on the task of picking up the treasure while performance of COCOA-reward noticeably suffers. Note that the performance of all methods has slightly decreased compared to Fig. 3, as the magnitude of the treasure reward is now smaller relative to the variance of the distractor rewards, resulting in a worse SNR for all methods.

5 Discussion

We present a theory for model-based credit assignment compatible with discrete actions and show in a comprehensive theoretical and experimental analysis that this yields a powerful policy gradient estimator, enabling long-term credit assignment by disentangling rewarding outcomes.

Building upon HCA [1], we focus on amortizing the estimation of the contribution coefficients in an inverse dynamics model, $p^\pi(a \mid s, u')$. The quality of this model is crucial for obtaining low-bias gradient estimates, but it is restricted to learn from on-policy data, and rewarding observations in case of $u = r$. Scaling these inverse models to complex environments will potentially exacerbate this tension, especially in sparse reward settings. A promising avenue for future work is to leverage forward dynamics models and directly estimate contribution coefficients from synthetic trajectories. While learning a forward model is a difficult problem in its own, its policy independence increases the data available for learning it. This would result in an algorithm close in spirit to Stochastic Value Gradients [20] and Dreamer [21–23] with the crucial advance that it enables model-based credit assignment on discrete actions. Another possibility to enable learning from non-rewarding observations is to learn a generative model that can recombine inverse models based on state representations into reward contributions (c.f. App. H).

Related work has explored the credit assignment problem through the lens of transporting rewards or value estimates towards previous states to bridge long-term dependencies [4, 5, 32, 35–41]. This approach is compatible with existing and well-established policy gradient estimators but determining how to redistribute rewards has relied on heuristic contribution analyses, such as via the access of memory states [4], linear decompositions of rewards [32, 35–39] or learned sequence models [5, 40, 41]. Leveraging our unbiased contribution analysis framework to reach more optimal reward transport is a promising direction for future research.

While we have demonstrated that contribution coefficients with respect to states as employed by HCA suffer from spurious contributions, any reward feature encoding that is fully predictive of the reward can in principle suffer from a similar problem in the case where each environment state has a unique reward value. In practice, this issue might occur in environments with continuous rewards. A potential remedy in this situation is to assume that the underlying reward distribution is stochastic, smoothing the contribution coefficients as now multiple states could have led to the same reward. This lowers the variance of the gradient estimator as we elaborate in App. G.

Finally, we note that our contribution coefficients are closely connected to causality theory [42] where the contribution coefficients correspond to performing *Do-interventions* on the causal graph to estimate their effect on future rewards (c.f. App I). Within causality theory, counterfactual reasoning goes a step further by inferring the external, uncontrollable environment influences and considering the consequences of counterfactual actions *given that all external influences remain the same* [3, 20, 42–44]. Extending COCOA towards this more advanced counterfactual setting by building upon recent work [3, 43] is an exciting direction for future research (c.f. App. I).

Concluding remarks. By overcoming the failure mode of *spurious contributions* in HCA, we have presented here a comprehensive theory on how to leverage model information for credit assignment, compatible with discrete action spaces. COCOA-reward and COCOA-feature are promising first algorithms in this framework, opening the way towards sample-efficient reinforcement learning by model-based credit assignment.

6 Acknowledgements

We thank Angelika Steger, Yassir Akram, Ida Momennejad, Blake Richards, Matt Botvinick and Joel Veness for discussions and feedback. Simon Schug is supported by the Swiss National Science Foundation (PZ00P3_186027). Seijin Kobayashi is supported by the Swiss National Science Foundation (CRSII5_173721). Simon Schug would like to kindly thank the TPU Research Cloud (TRC) program for providing access to Cloud TPUs from Google.

References

[1] Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Remi Munos. Hindsight Credit Assignment. In H. Wallach, H. Larochelle, A. Beygelzimer, F Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 12488–12497. Curran Associates, Inc., 2019.

[2] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning, second edition: An Introduction*. MIT Press, November 2018.

[3] Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, Marcus Hutter, Lars Buesing, and Rémi Munos. Counterfactual Credit Assignment in Model-Free Reinforcement Learning. In *Proceedings of the 38 th International Conference on Machine Learning, PMLR 139*, 2021.

[4] Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature Communications*, 10(1):5223, November 2019. Number: 1 Publisher: Nature Publishing Group.

[5] Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return Decomposition for Delayed Rewards. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13566–13577. Curran Associates, Inc., 2019.

[6] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992.

[7] Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning. *Journal of Machine Learning Research*, 5(Nov): 1471–1530, 2004.

[8] Théophane Weber, Nicolas Heess, Lars Buesing, and David Silver. Credit Assignment Techniques in Stochastic Computation Graphs. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 2650–2660. PMLR, April 2019. ISSN: 2640-3498.

[9] Jonathan Baxter and Peter L. Bartlett. Infinite-Horizon Policy-Gradient Estimation. *Journal of Artificial Intelligence Research*, 15:319–350, November 2001.

[10] Philip Thomas. Bias in Natural Actor-Critic Algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, pages 441–448. PMLR, January 2014. ISSN: 1938-7228.

[11] Sham Kakade. Optimizing Average Reward Using Discounted Rewards. In David Helmbold and Bob Williamson, editors, *Computational Learning Theory*, Lecture Notes in Computer Science, pages 605–615, Berlin, Heidelberg, 2001. Springer.

[12] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations*. arXiv, 2016. arXiv:1506.02438 [cs].

[13] Peter Marbach and John Tsitsiklis. Approximate Gradient Methods in Policy-Space Optimization of Markov Reward Processes. *Discrete Event Dynamic Systems*, page 38, 2003.

[14] OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning, December 2019. arXiv:1912.06680 [cs, stat].

[15] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik's Cube with a Robot Hand, October 2019. arXiv:1910.07113 [cs, stat].

[16] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, November 2019. Number: 7782 Publisher: Nature Publishing Group.

[17] Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, July 1991.

[18] Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-Efficient Reinforcement Learning with Stochastic Ensemble Value Expansion. In *32nd Conference on Neural Information Processing Systems*. arXiv, 2018. arXiv:1807.01675 [cs, stat].

[19] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-Ensemble Trust-Region Policy Optimization. 2018.

[20] Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning Continuous Control Policies by Stochastic Value Gradients. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[21] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, March 2020. Number: arXiv:1912.01603 arXiv:1912.01603 [cs].

[22] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with Discrete World Models. In *International Conference on Learning Representations*, 2021. Number: arXiv:2010.02193 arXiv:2010.02193 [cs, stat].

[23] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Domains through World Models, January 2023. arXiv:2301.04104 [cs, stat].

[24] Mikael Henaff, William F. Whitney, and Yann LeCun. Model-Based Planning with Discrete and Continuous Actions, April 2018. arXiv:1705.07177 [cs].

[25] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1928–1937. PMLR, June 2016. ISSN: 1938-7228.

[26] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems 12*, 1999.

[27] Chris Nota and Philip S Thomas. Is the Policy Gradient a Gradient? In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*, page 9, 2020.

[28] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-Dimensional Continuous Control Using Generalized Advantage Estimation, October 2018. arXiv:1506.02438 [cs].

[29] Vyacheslav Alipov, Riley Simmons-Edler, Nikita Putintsev, Pavel Kalinin, and Dmitry Vetrov. Towards Practical Credit Assignment for Deep Reinforcement Learning, February 2022. arXiv:2106.04499 [cs].

[30] Peter Dayan. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation*, 5(4):613–624, July 1993. Conference Name: Neural Computation.

[31] Tejas D. Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J. Gershman. Deep Successor Reinforcement Learning, June 2016. Number: arXiv:1606.02396 arXiv:1606.02396 [cs, stat].

[32] David Raposo, Sam Ritter, Adam Santoro, Greg Wayne, Theophane Weber, Matt Botvinick, Hado van Hasselt, and Francis Song. Synthetic Returns for Long-Term Credit Assignment. *arXiv:2102.12425 [cs]*, February 2021. arXiv: 2102.12425.

[33] John W. Roberts and Russ Tedrake. Signal-to-Noise Ratio Analysis of Policy Gradient Algorithms. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1361–1368. Curran Associates, Inc., 2009.

[34] Ching-An Cheng, Xinyan Yan, and Byron Boots. Trajectory-wise Control Variates for Variance Reduction in Policy Gradient Methods. In *Proceedings of the Conference on Robot Learning*, pages 1379–1394. PMLR, May 2020. ISSN: 2640-3498.

[35] Zhizhou Ren, Ruihan Guo, Yuan Zhou, and Jian Peng. Learning Long-Term Reward Redistribution via Randomized Return Decomposition. January 2022.

[36] Yonathan Efroni, Nadav Merlis, and Shie Mannor. Reinforcement Learning with Trajectory Feedback. In *The Thirty-Fifth AAAI Conference on Artificial Intelligenc*. arXiv, March 2021.

[37] Minah Seo, Luiz Felipe Vecchietti, Sangkeum Lee, and Dongsoo Har. Rewards Prediction-Based Credit Assignment for Reinforcement Learning With Sparse Binary Rewards. *IEEE ACCESS*, 7:118776–118791, 2019. Accepted: 2019-09-24T11:21:52Z Publisher: IEEE-INST ELECTRICAL ELECTRONICS ENGINEERS INC.

[38] Markel Sanz Ausin, Hamoon Azizsoltani, Song Ju, Yeo Jin Kim, and Min Chi. InferNet for Delayed Reinforcement Tasks: Addressing the Temporal Credit Assignment Problem. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 1337–1348, December 2021.

[39] Hamoon Azizsoltani, Yeo Jin Kim, Markel Sanz Ausin, Tiffany Barnes, and Min Chi. Unobserved Is Not Equal to Non-existent: Using Gaussian Processes to Infer Immediate Rewards Across Contexts. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 1974–1980, 2019.

[40] Vihang P. Patil, Markus Hofmarcher, Marius-Constantin Dinu, Matthias Dorfer, Patrick M. Blies, Johannes Brandstetter, Jose A. Arjona-Medina, and Sepp Hochreiter. Align-RUDDER: Learning From Few Demonstrations by Reward Redistribution. *Proceedings of Machine Learning Research*, 162, 2022. arXiv: 2009.14108.

[41] Johan Ferret, Raphaël Marinier, Matthieu Geist, and Olivier Pietquin. Self-Attentional Credit Assignment for Transfer in Reinforcement Learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 2655–2661, July 2020. arXiv:1907.08027 [cs].

[42] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. Adaptive Computation and Machine Learning. November 2018.

[43] Lars Buesing, Theophane Weber, Yori Zwols, Sebastien Racaniere, Arthur Guez, Jean-Baptiste Lespiau, and Nicolas Heess. Woulda, Coulda, Shoulda: Counterfactually-Guided Policy Search. In *International Conference on Learning Representations*, 2019. arXiv: 1811.06272.

[44] Ioana Bica, Ahmed M. Alaa, James Jordon, and Mihaela van der Schaar. Estimating Counterfactual Treatment Outcomes over Time Through Adversarially Balanced Representations. In *International Conference on Learning Representations*. arXiv, February 2020. arXiv:2002.04083 [cs, stat].

[45] Pushi Zhang, Li Zhao, Guoqing Liu, Jiang Bian, Minlie Huang, Tao Qin, and Tie-Yan Liu. Independence-aware Advantage Estimation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3349–3355, Montreal, Canada, August 2021. International Joint Conferences on Artificial Intelligence Organization.

[46] Kenny Young. Variance Reduced Advantage Estimation with δ Hindsight Credit Assignment. *arXiv:1911.08362 [cs]*, September 2020. arXiv: 1911.08362.

[47] Michel Ma and Bacon Pierre-Luc. Counterfactual Policy Evaluation and the Conditional Monte Carlo Method. In *Offline Reinforcement Learning Workshop, NeurIPS*, 2020.

[48] Paul Bratley, Bennett L. Fox, and Linus E. Schrage. *A Guide to Simulation*. Springer, New York, NY, 1987.

[49] J. M. Hammersley. Conditional Monte Carlo. *Journal of the ACM*, 3(2):73–76, April 1956.

[50] Dilip Arumugam, Peter Henderson, and Pierre-Luc Bacon. An Information-Theoretic Perspective on Credit Assignment in Reinforcement Learning. *arXiv:2103.06224 [cs, math]*, March 2021. arXiv: 2103.06224.

[51] Kenny Young. Hindsight Network Credit Assignment: Efficient Credit Assignment in Networks of Discrete Stochastic Units. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36 (8):8919–8926, June 2022. Number: 8.

[52] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E. Turner, and Sergey Levine. Q-Prop: Sample-Efficient Policy Gradient with An Off-Policy Critic. 2017.

[53] Philip S. Thomas and Emma Brunskill. Policy Gradient Methods for Reinforcement Learning with Function Approximation and Action-Dependent Baselines, June 2017. arXiv:1706.06643 [cs].

[54] Hao Liu*, Yihao Feng*, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent Control Variates for Policy Optimization via Stein Identity. February 2022.

[55] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M. Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance Reduction for Policy Gradient with Action-Dependent Factorized Baselines. February 2022.

[56] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard Turner, Zoubin Ghahramani, and Sergey Levine. The Mirage of Action-Dependent Baselines in Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5015–5024. PMLR, July 2018. ISSN: 2640-3498.

[57] Chris Nota, Philip Thomas, and Bruno C. Da Silva. Posterior Value Functions: Hindsight Baselines for Policy Gradient Methods. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8238–8247. PMLR, July 2021. ISSN: 2640-3498.

[58] Arthur Guez, Fabio Viola, Theophane Weber, Lars Buesing, Steven Kapturowski, Doina Precup, David Silver, and Nicolas Heess. Value-driven Hindsight Modelling. *Advances in Neural Information Processing Systems*, 33, 2020.

[59] David Venuto, Elaine Lau, Doina Precup, and Ofir Nachum. Policy Gradients Incorporating the Future. January 2022.

[60] Jiawei Huang and Nan Jiang. From Importance Sampling to Doubly Robust Policy Gradient. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4434–4443. PMLR, November 2020. ISSN: 2640-3498.

[61] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 278–287, San Francisco, CA, USA, June 1999. Morgan Kaufmann Publishers Inc.

[62] Jürgen Schmidhuber. Formal Theory of Creativity, Fun, and Intrinsic Motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, September 2010.

[63] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying Count-Based Exploration and Intrinsic Motivation. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[64] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. December 2018.

[65] Ofir Marom and Benjamin Rosman. Belief Reward Shaping in Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. Number: 1.

[66] Farzan Memarian, Wonjoon Goo, Rudolf Lioutikov, Scott Niekum, and Ufuk Topcu. Self-Supervised Online Reward Shaping in Sparse-Reward Environments, July 2021. arXiv:2103.04529 [cs].

[67] Halit Bener Suay, Tim Brys, Matthew E. Taylor, and Sonia Chernova. Learning from Demonstration for Shaping through Inverse Reinforcement Learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, pages 429–437, Richland, SC, May 2016. International Foundation for Autonomous Agents and Multiagent Systems.

[68] Yuchen Wu, Melissa Mozifian, and Florian Shkurti. Shaping Rewards for Reinforcement Learning with Imperfect Demonstrations using Generative Models, November 2020. arXiv:2011.01298 [cs].

[69] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient Estimation Using Stochastic Computation Graphs. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[70] Michel Ma, Pierluca D'Oro, Yoshua Bengio, and Pierre-Luc Bacon. Long-Term Credit Assignment via Model-based Temporal Shortcuts. In *Deep RL Workshop NeurIPS*, October 2021.

[71] Nan Rosemary Ke, Anirudh Goyal ALIAS PARTH GOYAL, Olexa Bilaniuk, Jonathan Binas, Michael C Mozer, Chris Pal, and Yoshua Bengio. Sparse Attentive Backtracking: Temporal Credit Assignment Through Reminding. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7640–7651. Curran Associates, Inc., 2018.

[72] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal Value Function Approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1312–1320. PMLR, June 2015. ISSN: 1938-7228.

[73] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[74] Paulo Rauber, Avinash Ummadisingu, Filipe Mutz, and Jürgen Schmidhuber. Hindsight policy gradients. December 2018.

[75] Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. Recall Traces: Backtracking Models for Efficient Reinforcement Learning. In *International Conference on Learning Representations*. arXiv, January 2019. arXiv:1804.00379 [cs, stat].

[76] Juergen Schmidhuber. Reinforcement Learning Upside Down: Don't Predict Rewards – Just Map Them to Actions, June 2020. arXiv:1912.02875 [cs].

[77] Rupesh Kumar Srivastava, Pranav Shyam, Filipe Mutz, Wojciech Jaśkowski, and Jürgen Schmidhuber. Training Agents using Upside-Down Reinforcement Learning, September 2021. arXiv:1912.02877 [cs].

[78] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision Transformer: Reinforcement Learning via Sequence Modeling. In *Advances in Neural Information Processing Systems*, volume 34, pages 15084–15097. Curran Associates, Inc., 2021.

[79] Michael Janner, Qiyang Li, and Sergey Levine. Offline Reinforcement Learning as One Big Sequence Modeling Problem. In *Advances in Neural Information Processing Systems*, volume 34, pages 1273–1286. Curran Associates, Inc., 2021.

[80] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

[81] Hado van Hasselt, Sephora Madjiheurem, Matteo Hessel, David Silver, André Barreto, and Diana Borsa. Expected Eligibility Traces. In *Association for the Advancement of Artificial Intelligence*. arXiv, February 2021. arXiv:2007.01839 [cs, stat].

[82] Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020.

[83] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep Variational Information Bottleneck. 2017.

[84] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, April 2000. arXiv:physics/0004057.

[85] Matthew Hausknecht and Peter Stone. Deep Recurrent Q-Learning for Partially Observable MDPs. In *Association for the Advancement of Artificial Intelligence*. arXiv, 2015. arXiv:1507.06527 [cs].

[86] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2555–2565. PMLR, May 2019. ISSN: 2640-3498.

[87] Karol Gregor, Danilo Jimenez Rezende, Frederic Besse, Yan Wu, Hamza Merzic, and Aaron van den Oord. Shaping Belief States with Generative Environment Models for RL. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*. arXiv, June 2019. Number: arXiv:1906.09237 arXiv:1906.09237 [cs, stat].

[88] Karol Gregor, George Papamakarios, Frederic Besse, Lars Buesing, and Theophane Weber. Temporal Difference Variational Auto-Encoder. 2019.

[89] Matthijs T J Spaan. Partially Observable Markov Decision Processes. *Reinforcement Learning*, page 27.

[90] K. J. Astrom. Optimal Control of Markov decision processes with incomplete state estimation. *J. Math. Anal. Applic.*, 10:174–205, 1965.

[91] Edward C. Tolman. Cognitive maps in rats and men. *Psychological Review*, 55:189–208, 1948. Place: US Publisher: American Psychological Association.

[92] Alexander A. Alemi, Ben Poole, Ian Fischer, Joshua V. Dillon, Rif A. Saurous, and Kevin Murphy. Fixing a Broken ELBO. In *Proceedings of the 35 th International Conference on Machine Learning,*. arXiv, February 2018. arXiv:1711.00464 [cs, stat].

[93] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. July 2022.

[94] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. arXiv, May 2014. Number: arXiv:1312.6114 arXiv:1312.6114 [cs, stat].

[95] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[96] Lukas Biewald. Experiment Tracking with Weights and Biases, 2020.

[97] Plotly Technologies Inc. Collaborative data science, 2015. Place: Montreal, QC Publisher: Plotly Technologies Inc.

$$= r(s,a) + \sum_{r' \in \mathcal{R}} \sum_{u' \in \mathcal{U}} \sum_{k=1}^{n-1} \gamma^k p^\pi(R_k = r', U_k = u' \mid s,a) r' + \tag{132}$$

$$\gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s,a) V^\pi(s') \tag{133}$$

$$= r(s,a) + \sum_{r' \in \mathcal{R}} \sum_{u' \in \mathcal{U}} \sum_{k=1}^{n-1} \gamma^k p^\pi(R' = r' \mid U' = u') p^\pi(U_k = u' \mid s,a) r' + \tag{134}$$

$$\gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s,a) V^\pi(s') \tag{135}$$

$$= r(s,a) + \sum_{u' \in \mathcal{U}} r(u') \sum_{k=1}^{n-1} \gamma^k p^\pi(U_k = u' \mid s,a) + \gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s,a) V^\pi(s') \tag{136}$$

$$= r(s,a) + \sum_{u' \in \mathcal{U}} r(u') \sum_{k=1}^{n-1} \gamma^k p^\pi(U_k = u' \mid s) \frac{\sum_{k'=1}^{n-1} \gamma^{k'} p^\pi(U_{k'} = u \mid s,a)}{\sum_{k'=1}^{n-1} \gamma^{k'} p^\pi(U_{k'} = u \mid s)} + \tag{137}$$

$$\gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s) \frac{p^\pi(S_n = s' \mid s,a)}{p^\pi(S_n = s' \mid s)} V^\pi(s') \tag{138}$$

$$= r(s,a) + \sum_{u' \in \mathcal{U}} r(u') \sum_{k=1}^{n-1} \gamma^k p^\pi(U_k = u' \mid s)(w_{n,\beta}(s,a,u') + 1) \tag{139}$$

$$+ \gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s)(w_n(s,a,s') + 1) V^\pi(s') \tag{140}$$

$$\tag{141}$$

where we use that U' is fully predictive of the reward R', and define $r(u') = \sum_{r' \in \mathcal{R}} p(R' = r' \mid U' = u') r'$ By subtracting the value function, we get

$$A(s,a) = r(s,a) - r^\pi(s) + \sum_{u' \in \mathcal{U}} r(u') \sum_{k=1}^{n-1} \gamma^k p^\pi(U_k = u' \mid s) w_{n,\beta}(s,a,u') \tag{142}$$

$$+ \gamma^n \sum_{s' \in \mathcal{S}} p^\pi(S_n = s' \mid s) w_n(s,a,s') V^\pi(s') \tag{143}$$

$$= r(s,a) - r^\pi(s) + \mathbb{E}_{\mathcal{T}(s,\pi)} \left[\sum_{k=1}^{n-1} \gamma^k w_{n,\beta}(s,a,U_k) R_k + \gamma^n w_n(s,a,S_n) V^\pi(S_n) \right] \tag{144}$$

□

Finally, we can sample from this advantage function to obtain an n-step COCOA gradient estimator, akin to Theorem 1.

Note that we require to learn the state-based contribution coefficients $w_n(s,a,s')$ to bootstrap the value function into the n-step return, as the value function requires a Markov state s' as input instead of a rewarding outcome encoding u'. Unfortunately, these state-based contribution coefficients will suffer from spurious contributions, akin to HCA, introducing a significant amount of variance into the n-step COCOA gradient estimator. We leave it to future research to investigate whether we can incorporate value functions into an n-step return, while using rewarding-outcome contribution coefficients $w(s,a,u')$ instead of state-based contribution coefficients $w_n(s,a,s')$.

Learning the contribution coefficients. We can learn the contribution coefficients $w_{\beta,n}(s,a,u')$ with the same strategies as described in Section 3, but now with training data from n-step trajectories instead of complete trajectories. If we use a discount $\gamma \neq 1$, we need to take this discount factor into account in the training distribution or loss function (c.f. App. J).

Correction to Theorem 7 of Harutyunyan et al. [1]. Harutyunyan et al. [1] propose a theorem similar to Theorem 10, with two important differences. The first one concerns the distribution on K in the graphical model of Fig. 15a. Harutyunyan et al. [1] implicitly use this graphical model, but with a different prior probability distribution on K:

$$p^{HCA}_{n,\beta}(K=k) = \begin{cases} \beta^{k-1}(1-\beta) & \text{if } 1 \leq k \leq n-1 \\ \beta^{n-1} & \text{if } k = n \\ 0 & \text{else} \end{cases} \tag{145}$$

The graphical model combined with the distribution on K defines the hindsight distribution $p^{\pi}_{n,\beta,HCA}(A = a \mid S = s, S' = s')$. The second difference is the specific Q-value estimator Harutyunyan et al. [1] propose. They use the hindsight distribution $p^{\pi}_{n,\beta,HCA}(A = a \mid S = s, S' = s')$ in front of the value function (c.f. Theorem 10), which considers that s' can be reached at any time step $k \sim p^{HCA}_{n,\beta}(k)$, whereas Theorem 10 uses $w_n(s, a, s')$ which considers that s' is reached exactly at time step $k = n$.

To the best of our knowledge, there is an error in the proposed proof of Theorem 7 by Harutyunyan et al. [1] for which we could not find a simple fix. For the interested reader, we briefly explain the error. One indication of the problem is that for $\beta \to 1$, all the probability mass of $p^{HCA}_{n,\beta}(K = k)$ is concentrated at $k = n$, hence the corresponding hindsight distribution $p^{\pi}_{n,\beta,HCA}(A = a \mid S = s, S' = s')$ considers only hindsight states s' encountered at time $k = n$. While this is not a mathematical error, it does not correspond to the intuition of a 'time independent hindsight distribution' the authors provide. In the proof itself, a conditional independence relation is assumed that does not hold. The authors introduce a helper variable Z defined on the state space $\mathcal{S}$, with a conditional distribution

$$\mu_k(Z = z \mid S' = s') = \begin{cases} \delta(z = s') & \text{if } 1 \leq k \leq n-1 \\ \tilde{d}^{\pi}(z \mid s') & \text{if } k = n \end{cases} \tag{146}$$

with the normalized discounted visit distribution $\tilde{d}^{\pi}(z \mid s') = (1-\gamma)\sum_k \gamma^k p^{\pi}(S_k = z \mid S_0 = s)$. We can model this setting as the graphical model visualized in Fig. 15b. In the proof (last line on page 15 in the supplementary materials of Harutyunyan et al. [1]), the following conditional independence is used:

$$p^{\pi}(A_0 = a \mid S_0 = s, S' = s', Z = z) = p^{\pi}(A_0 = a \mid S_0 = s, S' = s') \tag{147}$$

However, Fig. 15b shows that S' is a collider on the path $A_0 \to S' \leftarrow K \to Z$. Hence, by conditioning on S' we open this collider path, making A_0 dependent on Z conditioned on S_0 and S', thereby invalidating the assumed conditional independence. For example, if Z is different from S', we know that $K = n$ (c.f. Eq. 146), hence Z can contain information about action A_0, beyond S', as S' ignores at which point in time s' is encountered.

L HCA-return is a biased estimator in many relevant environments

L.1 HCA-return

Besides HCA-state, Harutyunyan et al. [1] introduced HCA-return, a policy gradient estimator that leverages the hindsight distribution conditioned on the return:

$$\sum_{t \geq 0} \nabla_\theta \log \pi(A_t \mid S_t) \Big(1 - \frac{\pi(A_t \mid S_t)}{p^{\pi}(A_t \mid S_t, Z_t)}\Big) Z_t \tag{148}$$

When comparing this estimator with COCOA-reward, we see two important differences: (i) HCA-return uses a hindsight function conditioned on the return instead of individual rewards, and (ii) HCA-return leverages the hindsight function as an action-dependent baseline for a Monte Carlo policy gradient estimate, instead of using it for contribution coefficients to evaluate counterfactual actions. Importantly, the latter difference causes the HCA-return estimator to be biased in many environments of relevance, even when using the ground-truth hindsight distribution.

L.2 HCA-return can be biased

An important drawback of HCA-return is that it can be biased, even when using the ground-truth hindsight distribution. Theorem 2 of Harutyunyan et al. 2019, considering the unbiasedness HCA-return, is valid under the assumption that for any possible random return Z for all possible trajectories

starting from state s, it holds that $p^\pi(a \mid s, z) > 0$. This restrictive assumption requires that for each observed state-action pair (s_t, a_t) along a trajectory, all counterfactual returns Z resulting from a counterfactual trajectory starting from s_t (not including a_t) result in $p^\pi(a_t \mid s_t, Z) > 0$. This implies that all returns (or rewarding states) reachable from s_t should also be reachable from (s_t, a_t).

Consider the following bandit setting as a simple example where the above assumption is not satisfied. The bandit has two arms, with a reward of 1 and -2, and a policy probability of $\frac{2}{3}$ and $\frac{1}{3}$ respectively. The advantage for both arms is 1 and -2. Applying eq. 6 from Harutyunyan et al. results in $A^\pi(s, a_1) = (1 - \frac{2}{3}) = \frac{1}{3}$ and $A^\pi(s, a_2) = -2(1 - 1/3) = -4/3$. This shows that the needed assumptions for an unbiased HCA-return estimator can be violated even in simple bandit settings.

M Additional details

M.1 Author contributions

This paper was a collaborative effort of all shared first authors working closely together. To do this fact better justice we give an idea of individual contributions in the following.

Alexander Meulemans*. Original idea, conceptualizing the theory and proving the theorems, conceptual development of the algorithms, experiment design, implementation of main method and environments, debugging, neural network architecture design, running experiments, connecting the project to existing literature, writing of manuscript, first draft and supplementary materials, feedback to the figures.

Simon Schug*. Conceptual development of the algorithms, experiment design, implementation of main method, baselines and environments, neural network architecture design, debugging, tuning and running experiments, writing of manuscript, creation of figures, writing of supplementary materials.

Seijin Kobayashi*. Conceptual development of the algorithms, experiment design, implementation of environments, baselines, main method and Dynamic Programming-based ground-truth methods, debugging, tuning and running experiments, feedback to the manuscript, writing of supplementary materials.

Nathaniel Daw. Regular project meetings, conceptual input and feedback for method and experimental design, connecting the project to existing literature, feedback to the manuscript and figures.

Gregory Wayne. Senior project supervision, conceptualising of the project idea, conceptual development of the algorithms, regular project meetings, technical and conceptual feedback for method and experimental design, connecting the project to existing literature, feedback to the manuscript and figures.

M.2 Compute resources

We used Linux workstations with Nvidia RTX 2080 and Nvidia RTX 3090 GPUs for development and conducted hyperparameter searches and experiments using 5 TPUv2-8, 5 TPUv3-8 and 1 Linux server with 8 Nvidia RTX 3090 GPUs over the course of 9 months. All of the final experiments presented take less than a few hours to complete using a single Nvidia RTX 3090 GPU. In total, we spent an estimated amount of 2 GPU months.

M.3 Software and libraries

For the results produced in this paper we relied on free and open-source software. We implemented our experiments in Python using JAX [95, Apache License 2.0] and the Deepmind Jax Ecosystem [82, Apache License 2.0]. For experiment tracking we used wandb [96, MIT license] and for the generation of plots we used plotly [97, MIT license].