Counterfactual Credit Assignment in Model-Free Reinforcement Learning

Thomas Mesnard [*1] **Théophane Weber** [*1] **Fabio Viola** [1] **Shantanu Thakoor** [1] **Alaa Saade** [1]
Anna Harutyunyan [1] **Will Dabney** [1] **Tom Stepleton** [1] **Nicolas Heess** [1] **Arthur Guez** [1] **Éric Moulines** [2]
Marcus Hutter [1] **Lars Buesing** [1] **Rémi Munos** [1]

Abstract

Credit assignment in reinforcement learning is the problem of measuring an action's influence on future rewards. In particular, this requires separating *skill* from *luck*, i.e. disentangling the effect of an action on rewards from that of external factors and subsequent actions. To achieve this, we adapt the notion of counterfactuals from causality theory to a model-free RL setup. The key idea is to condition value functions on *future* events, by learning to extract relevant information from a trajectory. We formulate a family of policy gradient algorithms that use these future-conditional value functions as baselines or critics, and show that they are provably low variance. To avoid the potential bias from conditioning on future information, we constrain the hindsight information to not contain information about the agent's actions. We demonstrate the efficacy and validity of our algorithm on a number of illustrative and challenging problems.

1. Introduction

Reinforcement learning (RL) agents act in their environments and learn to achieve desirable outcomes by maximizing a reward signal. A key difficulty is the problem of *credit assignment* (Minsky, 1961), i.e. to understand the relation between actions and outcomes, and to determine to what extent an outcome was caused by external, uncontrollable factors. In doing so we aim to disentangle the relative aspects of 'skill' and 'luck' in an agent's performance. One possible solution to this problem is for the agent to build a model of the environment, and use it to obtain a more fine-grained understanding of the effects of an action. While this topic has recently generated a lot of interest (Heess et al., 2015; Ha & Schmidhuber, 2018; Hamrick, 2019; Kaiser et al., 2019; Schrittwieser et al., 2019), it remains difficult to model complex, partially observed environments.

In contrast, model-free reinforcement learning algorithms such as policy gradient methods (Williams, 1992; Sutton et al., 2000) perform simple time-based credit assignment, where events and rewards happening after an action are credited to that action, *post hoc ergo propter hoc*. While unbiased in expectation, this coarse-grained credit assignment typically has high variance, and the agent will require a large amount of experience to learn the correct relation between actions and rewards. Another issue is that existing model-free methods are not capable of *counterfactual reasoning*, i.e. reasoning about what would have happened had different actions been taken *with everything else remaining the same*. Given a trajectory, model-free methods can in fact only learn about the actions that were actually taken to produce the data, and this limits the ability of the agent to learn efficiently.

As environments grow in complexity due to partial observability, scale, long time horizons, and increasing number of agents, actions taken by an agent will only affect a vanishing part of the outcome, making it increasingly difficult to learn from classical reinforcement learning algorithms. We need better credit assignment techniques.

In this paper, we investigate a new method of credit assignment for model-free reinforcement learning which we call *Counterfactual Credit Assignment* (CCA). CCA leverages *hindsight* information to implicitly perform counterfactual evaluation–an estimate of the return for actions other than the ones which were chosen. These counterfactual returns can be used to form unbiased and lower variance estimates of the policy gradient by building future-conditional baselines. Unlike classical Q functions, which also provide an estimate of the return for all actions but do so by averaging over all possible futures, our methods provide trajectory-specific counterfactual estimates, i.e. an estimate of the return for different actions, but keeping as many of the ex-

[*] Equal contribution [1] DeepMind [2] INRIA XPOP, CMAP, École Polytechnique, Palaiseau, France. Correspondence to: Théophane Weber <theophane@deepmind.com>, Thomas Mesnard <mesnard@deepmind.com>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

ternal factors constant between the return and its counterfactual estimate[1]. Such a method would perform finer-grained credit assignment and could greatly improve data efficiency in environments with complex credit assignment structures. Our method is inspired by ideas from causality theory, but does not require learning a model of the environment.

Our main contributions are: a) introducing a family of novel policy gradient estimators that leverage hindsight information and generalizes previous approaches, b) proposing a practical instantiation of this algorithm with sufficiency conditions for unbiasedness and guarantees for lower variance, c) introducing a set of environments which further our understanding of when credit assignment is made difficult due to exogenous noise, long-term effects and task interleaving, and thus leads to poor policy learning, d) demonstrating the improved performance of our algorithm on these environments, e) formally connecting our results to notions of counterfactuals in causality theory, further linking the causal inference and reinforcement learning literatures.

2. Counterfactual Credit Assignment

2.1. Notation

We use capital letters for random variables and lowercase for the value they take. Consider a generic MDP $(\mathcal{X}, \mathcal{A}, p, r, \gamma)$. Given a current state $x \in \mathcal{X}$ and assuming an agent takes action $a \in \mathcal{A}$, the agent receives reward $r(x, a)$ and transitions to a state $y \sim p(\cdot|x, a)$. The state (resp. action, reward) of the agent at step t is denoted X_t (resp. A_t, R_t). The initial state of the agent X_0 is a fixed x_0. The agent acts according to a policy π, i.e. action A_t is sampled from the policy $\pi_\theta(\cdot|X_t)$ where θ are the policy parameters, and aims to optimize the expected discounted return $\mathbb{E}[G] = \mathbb{E}[\sum_t \gamma^t R_t]$. The return G_t from step t is $G_t = \sum_{t' \geq t} \gamma^{t'-t} R_{t'}$. Note $G = G_0$. Finally, we define the score function $s_\theta(\pi_\theta, a, x) = \nabla_\theta \log \pi_\theta(a|x)$; the score function at time t is denoted $S_t = \nabla_\theta \log \pi_\theta(A_t|X_t)$. In the case of a partially observed environment, we assume the agent receives an observation E_t at every time step, and simply define X_t to be the set of all previous observations, actions and rewards $X_t = (O_{\leq t})$, with $O_t = (E_t, A_{t-1}, R_{t-1})$.[2] $\mathbb{P}(X)$ will denote the probability distribution of a random variable X.

2.2. Policy gradient algorithms

We begin by recalling two forms of policy gradient algorithms and the credit assignment assumptions they make. The first is the REINFORCE algorithm introduced by Williams (1992), which we will also call the single-action policy gradient estimator. The gradient of $\mathbb{E}[G]$ is given by:

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\Big[\sum_{t \geq 0} \gamma^t \, S_t \, \left(G_t - V(X_t)\right)\Big], \qquad (1)$$

where $V(X_t) = \mathbb{E}[G_t|X_t]$. Let's note here that $V(X_t)$ (resp. $Q(X_t, A_t) = \mathbb{E}[G_t|X_t, A_t]$) is the value function (resp. Q-function) for the policy π_θ but for notation simplicity the dependence on the policy will be implicit through the rest of this paper.

The appeal of this estimator lies in its simplicity and generality: to evaluate it, the only requirement is the ability to simulate trajectories, and compute both the score function and the return.

Note that subtracting the value function $V(X_t)$ from the return G_t does not bias the estimator and typically reduces variance, since the resulting estimate makes an action A_t more likely proportionally not to the return, but to which extent the return was higher than what was expected before the action was taken (Williams, 1992). Such a function will be called a *baseline* in the following. In theory, the baseline can be any function of X_t. It is however typically assumed that it does not depend on any variable 'from the future' (including the action about to be taken, A_t), i.e. with time index greater than t, since including variables which are (causally) affected by the action generally results in a biased estimator (Weber et al., 2019).

This estimator updates the policy through the score term; note however the learning signal only updates the policy $\pi_\theta(a|X_t)$ for the taken action $A_t = a$ (other actions are only updated through normalization of action probabilities). The policy gradient theorem from Sutton et al. (2000), which we will also call all-action policy gradient, shows it is possible to provide learning signal to all actions, given we have access to a Q-function, $Q(x, a) = \mathbb{E}[G_t|X_t = x, A_t = a]$, which we will call a *critic* in the following. The gradient of $\mathbb{E}[G]$ is given by:

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\Big[\sum_{t \geq 0} \gamma^t \sum_{a \in \mathcal{A}} \nabla_\theta \pi(a|X_t) Q(X_t, a)\Big]. \qquad (2)$$

2.3. Intuitive example on hindsight reasoning and skill versus luck

Imagine a scenario in which Alice just moved to a new city, is learning to play soccer, and goes to the local soccer field to play a friendly game with a group of other kids she has never met. As the game goes on, Alice does not seem to play at her best and makes some mistakes. It turns out however her partner Megan is a strong player, and eventually scores the goal that makes the game a victory. What should Alice learn from this game?

When using the single-action policy gradient estimate, the

[1] From from a causality standpoint, one-step action-value functions are interventional concepts ("What would happen if") instead of counterfactuals ("What would have happened if").

[2] Previous actions and rewards are provided as part of the observation as it is generally beneficial to do so in partially observable Markov decision processes.

outcome of the game being a victory, and assuming a ± 1 reward scheme, all her actions taken during the game are made more likely; this is in spite of the fact that during this particular game she may not have played well and that the victory is actually due to her strong teammate. From an RL point of view, her actions are wrongly credited for the victory and positively reinforced as a result; effectively, Alice was lucky rather than skillful. Regular baselines do not mitigate this issue, as Alice did not a priori know the skill of Megan, resulting in an assumption that Megan was of average strength and therefore a guess that their team had a 50% chance of winning. This could be fixed by understanding that Megan's strong play were not a consequence of Alice's play, that her skill was a priori unknown but known in hindsight, and that it is therefore valid to retroactively include her skill level in the baseline. A hindsight baseline, conditioned on Megan's estimated skill level, would therefore be closer to 1, driving the advantage estimate (and corresponding learning signal) close to 0.

As pointed out by Buesing et al. (2019), situations in which hindsight information is helpful in understanding a trajectory are frequent. In that work, the authors adopt a model-based framework, where hindsight information is used to ground counterfactual trajectories (i.e. trajectories under *different actions, but same randomness*). Our proposed approach follows a similar intuition, but is model-free: we attempt to *measure*—instead of model— information known in hindsight to compute a *future-conditional baseline*, but in a way that maintains unbiasedness. As we will see later, this corresponds to a constraint that the captured information must not have been caused by the agent.

2.4. Future-conditional (FC-PG) and Counterfactual (CCA-PG) Policy Gradient Estimators

Intuitively, our approach for assigning proper credit to action A_t relies on measuring statistics Φ_t that capture relevant information from the trajectory (e.g. including observations $O_{t'}$ at times t' greater than t). We then learn value functions or critics which are conditioned on the additional hindsight information contained in Φ_t. In general, these future-conditional values and critics would be biased for use in a policy gradient algorithm; we therefore use an importance correction term to eliminate this bias.

Theorem 1 (Future-Conditional Policy Gradient (FC-PG) estimators)**.** *Let Φ_t be an arbitrary random variable. Assuming that $\frac{\pi(a|X_t)}{\mathbb{P}(a|X_t,\Phi_t)} < \infty$ for all a, the following is the single-action unbiased estimator of the gradient of* $\mathbb{E}[G]$*:*

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\Big[\sum_t \gamma^t \, S_t\Big(G_t - \frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t,\Phi_t)} V(X_t,\Phi_t)\Big)\Big] \tag{3}$$

where $V(x,\phi) = \mathbb{E}[G_t|X_t = x, \Phi_t = \phi]$ is the future Φ-conditional value function .

With no requirements on Φ_t, we also have an all-action unbiased estimator:

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\big[\sum_{t,a} \gamma^t \nabla_\theta \log \pi(a|X_t) \mathbb{P}(a|X_t,\Phi_t) Q(X_t,\Phi_t,a)\big]$$

where $Q(x,\phi,a) = \mathbb{E}[G_t|X_t = x, \Phi_t = \phi, A_t = a]$ is the future-conditional Q function (critic). Furthermore, we have $Q(X_t,a) = \mathbb{E}\left[Q(X_t,\Phi_t,a)\frac{\mathbb{P}(a|X_t,\Phi_t)}{\pi(a|X_t)}\right]$.

Intuitively, the $\frac{\pi(a|X_t)}{\mathbb{P}(a|X_t,\Phi_t)} < \infty$ condition means that knowing Φ_t should not preclude any action a which was possible for π from having potentially produced Φ_t. A counterexample is $\Phi_t = A_t$; knowing Φ_t precludes any action $a \neq A_t$ from having produced Φ_t. Typically, Φ_t will be chosen to a function of the present and future trajectory $(X_s, A_s, R_s)_{s \geq t}$. The estimators above are very general and generalize similar estimators (HCA) introduced by Harutyunyan et al. (2019) (see App. C for a discussion of how HCA can be rederived from FC-PG) and different choices of Φ will have varying properties. Φ may be hand-crafted using domain knowledge, or, as we will see later, learned using appropriate objectives. Note that in general an FC-PG estimator doesn't necessarily have lower variance (a good proxy for fine-grained credit assignment) than the classical policy gradient estimator; this is due to the variance introduced by the importance weighting scheme. It would be natural to study an estimator where this effect is nullified through independence of the action and statistics Φ (resulting in a ratio of 1).

The resulting advantage estimate could thus be interpreted not just as an estimate of 'what outcome should I expect', but also a measure of 'how (un)lucky did I get?' and 'what other outcomes might have been possible in this precise situation, had I acted differently'. It will in turn provide finer-grained credit for action A_t in a sense to be made precise below.

Corollary 1 (Counterfactual Policy Gradient (CCA-PG))**.** *If A_t is independent from Φ_t given X_t, the following is an unbiased* single-action *estimator of the gradient of* $\mathbb{E}[G]$*:*

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\left[\sum_t \gamma^t \, S_t \left(G_t - V(X_t,\Phi_t)\right)\right]. \tag{4}$$

Furthermore, the hindsight advantage estimate has no higher variance than the forward one:

$$\mathbb{E}\left[\left(G_t - V(X_t,\Phi_t)\right)^2\right] \leq \mathbb{E}\left[\left(G_t - V(X_t)\right)^2\right].$$

Similarly, for the all-action *estimator:*

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\Big[\sum_t \gamma^t \sum_a \nabla_\theta \pi(a|X_t) Q(X_t,\Phi_t,a)\Big]. \tag{5}$$

Also, we have for all a,

$$Q(X_t,a) = \mathbb{E}[Q(X_t,\Phi_t,a)|X_t, A_t = a]$$

The benefit of the first estimator (equation 4) is clear: under the specified condition, and compared to the regular policy gradient estimator, the CCA estimator also has no bias, but the variance of its advantage estimate $G_t - V(X_t, \Phi_t)$ (the critical component behind variance of the overall estimator) is no higher.

For the all-action estimator, the benefits of CCA (equation 5) are less self-evident, since this estimator has *higher* variance than the regular all action estimator (which has variance 0). The interest here lies in bias due to learning imperfect Q functions. Both estimators require learning a Q function from data; any error in Q leads to a bias in π. Learning $Q(X_t, a)$ requires averaging over all possible trajectories initialized with state X_t and action a: in high variance situations, this will require a lot of data. In contrast, $Q(X_t, \Phi_t, a)$ predicts the average of the return G_t *conditional* on (X_t, Φ_t, a); if Φ_t has a high impact on G_t, the variance of that conditional return will be lower, and learning its average will in turn be far easier and data efficient.

2.5. Learning the relevant statistics: practical implementation of CCA-PG

The previous section proposes a sufficient condition on Φ for useful estimators to be derived. A question remains - how to compute such a Φ from the trajectory? While useful Φ could be handcrafted using expert knowledge, we propose to *learn* to extract Φ from the trajectory. The learning signal will be guided by two objectives: first, we will encourage Φ_t to be conditionally independent from A_t, as it is required for the estimator to be valid. Second, corollary 1 highlights that hindsight features which are predictive of the return lead to a decreased variance of the advantage estimate. To summarize, we want Φ to be predictive of the return while being independent of the action being currently credited. The corresponding hindsight conditional baseline would capture the 'luck' part of the outcome while the advantage estimate would capture the 'skill' aspect of it. We detail our agent components and losses below. See also Fig. 1 for a depiction of the resulting architecture and Appendix A for more details.

Agent components:

- **Agent network**: Our algorithm can generally be applied to arbitrary environments (e.g. POMDPs), so we assume the agent constructs an internal state X_t from past observations $(O_{t'})_{t' \leq t}$ using an arbitrary network, for instance an RNN, i.e. $X_t = \text{RNN}_{\theta_{\text{fs}}}(O_t, X_{t-1})$[3]. From X_t the agent computes a policy $\pi_{\theta_{\text{fs}}}(a|X_t)$, where θ_{fs} denotes the parameters of the representation network and policy.
- **Hindsight network**: Additionally, we assume the agent uses a hindsight network φ with parameters θ_{hs} which computes a hindsight statistic $\Phi_t = \varphi((X, A, R))$ which may depend arbitrarily on the vectors of observations, agent states and actions (in particular, it may depend on observations from timesteps $t' \geq t$).
- **Value network**: The third component is a future-conditional value network $V_{\theta_{\text{V}}}(X_t, \Phi_t)$, with parameters θ_{V}.
- **Hindsight predictor**: The last component is a probabilistic predictor h_ω with parameters ω that takes X_t, Φ_t as input and outputs a distribution over A_t which is used to enforce the independence condition.

Learning objectives:

- The first loss is the hindsight baseline loss $\mathcal{L}_{\text{hs}} = \sum_t (G_t - V_{\theta_{\text{V}}}(X_t, \Phi_t))^2$.
- The second loss is the independence loss, which ensures the conditional independence between A_t and Φ_t. There exists multiple ways to measure dependence between random variables; we assume a surrogate *independence maximization* (IM) loss $\mathcal{L}_{\text{IM}}(X_t)$ which is non-negative and zero if and only if A_t and Φ_t are conditionally independent given X_t. An example is to choose the Kullback-Leibler divergence between the distributions $\mathbb{P}(A_t|X_t)$ and $\mathbb{P}(A_t|X_t, \Phi_t)$. In this case, the KL can be estimated by $\sum_a \mathbb{P}(a|X_t)\left(\log \mathbb{P}(a|X_t) - \log \mathbb{P}(a|X_t, \Phi_t)\right)$; $\log \mathbb{P}(a|X_t)$ is simply the policy $\pi(a|X_t)$; the posterior $\mathbb{P}(a|X_t, \Phi_t)$ is generally not known exactly, but we estimate it with the probabilistic predictor $h_\omega(A_t|X_t, \Phi_t)$, which we train with the next loss.
- The third loss is the hindsight predictor loss, which we train by minimizing the supervised learning loss $\mathcal{L}_{\text{sup}} = -\sum_t \mathbb{E}[\log h_\omega(A_t|X_t, \Phi_t)]$ on samples (X_t, A_t, Φ_t) from the trajectory (note that this is a proper scoring rule, i.e. the optimal solution to the loss is the true probability $\mathbb{P}(a|X_t, \Phi_t)$).
- The last loss is the policy gradients surrogate objective, implemented as $\mathcal{L}_{\text{PG}} = \sum_t \log \pi_\theta(A_t|X_t)\overline{(G_t - V(X_t, \Phi_t))}$, where the bar notation indicates that the quantity is treated as a constant from the point of view of gradient computation, as is standard.

The overall loss is therefore $\mathcal{L} = \mathcal{L}_{\text{PG}} + \lambda_{\text{hs}}\mathcal{L}_{\text{hs}} + \lambda_{\text{sup}}\mathcal{L}_{\text{sup}} + \lambda_{\text{IM}}\mathcal{L}_{\text{IM}}$. We again want to highlight the very special role played by ω here: only $\mathcal{L}_{\text{sup}}$ is optimized with respect to ω (the parameters of the probabilistic predictor), while all the other losses are optimized treating ω as a constant.

[3] Obviously, if the environment is fully observed, a feed-forward network suffices.

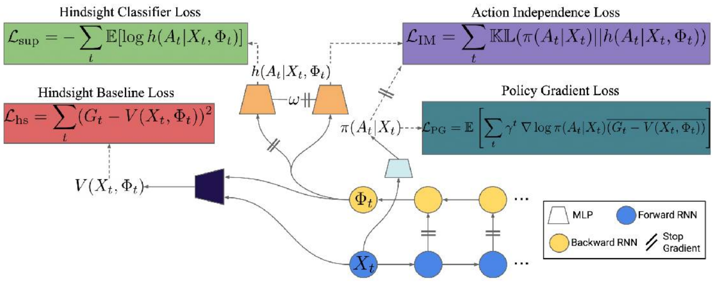
Hindsight Classifier Loss
$\mathcal{L}_{\text{sup}} = -\sum_t \mathbb{E}[\log h(A_t|X_t, \Phi_t)]$
Action Independence Loss
$\mathcal{L}_{\text{IM}} = \sum_t \mathbb{KL}(\pi(A_t|X_t)||h(A_t|X_t, \Phi_t))$
$h(A_t|X_t, \Phi_t)$
Hindsight Baseline Loss
$\mathcal{L}_{\text{hs}} = \sum_t (G_t - V(X_t, \Phi_t))^2$
ω
Policy Gradient Loss
$\pi(A_t|X_t)$
$\mathcal{L}_{\text{PG}} = \mathbb{E}\left[\sum_t \gamma^t \nabla \log \pi(A_t|X_t)\overline{(G_t - V(X_t, \Phi_t))}\right]$
$V(X_t, \Phi_t)$
Φ_t
X_t
...
MLP
Forward RNN
Backward RNN
Stop Gradient

Figure 1: Counterfactual Credit Assignment in a nutshell: (1) The backward RNN which in this example computes the hindsight features is shaped by the hindsight baseline loss. This ensures that it is predictive of the return. (2) However, to have an unbiased baseline, this hindsight feature Φ_t needs to be independent from the action A_t. To that end, we first train a hindsight predictor that tries to predict what action has been taken a time t from X_t and Φ_t. (3) Then the action independence loss helps removing any information about A_t from the hindsight feature Φ_t (This only enforces that the output of the backward RNN Φ_t is independent of the action A_t. However, this could potentially translate in Φ_t being independent from further actions). This loss only impacts the backward RNN and no gradient is being applied to the hindsight predictor MLP. (4) Finally, the policy gradient loss helps improving the policy while no gradient is being sent to the hindsight baseline (i.e as expressed by the bar notation).

3. Connections to causality

In this section we provide a formal connection between the CCA-PG estimator and counterfactuals in causality theory (this connection is investigated in greater depth in appendices F and G).

To this end, we assume that the MDP $(\mathcal{X}, \mathcal{A}, p, r, \gamma)$ in question is generated by an underlying structural causal models (SCM) analogous to (Buesing et al., 2019; Zhang, 2020). In this setting the trajectory $(X_s, A_s, R_s)_{s \geq t}$ and return G_t resulting from the agent-environment interaction is represented as the output of a deterministic function f taking as input the current state X_t, the action A_t, and a set of exogenous random variables $\boldsymbol{\mathcal{E}}$ which do not have any causal ancestors (in the graph). The latter represent the randomness required for sampling all future actions, transitions, and rewards. Such a "reparametrization" of trajectories and return is always possible, i.e. there is always an SCM (possibly non unique) that induces the same joint distribution $\mathbb{P}$ as the original MDP. Intuitively, $\boldsymbol{\mathcal{E}}$ represent all factors external to A_t which affect the outcome[4].

SCMs allow to formally define the notion of counterfactual. Given an observed trajectory $\tau = (X_s, A_s, R_s)_{s \geq t}$, we define the counterfactual trajectory τ' for an alternative action $A'_t = a'_t$ as a the output of the following procedure:

- Abduction: infer the exogenous noise variables ϵ under the factual observation: $\epsilon \sim \mathbb{P}(\mathcal{E}|\tau)$.
- Intervention: Fix the value of A'_t to a'_t (mutilating incoming causal arrows).
- Prediction: Evaluate the counterfactual outcome τ' conditional on the fixed values $\mathcal{E}$ and $A_t = a'_t$ yielding $\tau' = f(x_t, a'_t, \epsilon)$

The counterfactual distribution will be denoted $P(\tau'|\text{observe}(\tau), \text{do}(A'_t = a'_t))$. Note that it typically requires knowledge of the model (SCM) to be computed; samples from the models which do not expose the exogenous variables $\boldsymbol{\mathcal{E}}$ are not typically not sufficient to identify the SCM, as several SCMs may correspond to the same distribution. However, under the CCA assumptions and an additional faithfulness assumption, we can show that the counterfactual return is indeed identifiable and is equal to the future conditional state-action value function:

Theorem 2. *Assume the causal model is faithful (i.e. that conditional independence assumptions are reflected in the graph structure and not only in the parameters). If Φ_t is conditionally independent from A_t given X_t, then the counterfactual distribution, having observed only Φ_t, is identifiable from samples of (X_t, Φ_t, A_t), and we have*

$$\mathbb{E}[G(\tau')|\tau' \sim P(\tau'|X_t = x, \textit{observe}(\Phi_t = \phi), \textit{do}(A'_t = a)] = Q(X_t = x, A_t = a, \Phi_t = \phi) \quad (6)$$

4. Numerical experiments

Given its guarantees on lower variance and unbiasedness, we run all our experiments on the single action version of CCA-PG and leave the all-action version for future work. We first investigate a bandit with feedback task, then a task that requires short and long-term credit assignment (i.e. Key-to-Door), and finally an interleaved multi-task setup where each episode is composed of randomly sampled and interleaved tasks. All results for Key-to-Door and interleaved multi-task are reported as median performances over 10 seeds with quartiles represented by a shaded area.

4.1. Bandit with Feedback

We first demonstrate the benefits of hindsight value functions in a toy problem designed to highlight these. We consider a contextual bandit problem with feedback. At each time step, the agent receives a context $-N \leq C \leq N$ (where N is an environment parameter), and based on the context, chooses an action $-N \leq A \leq N$. The agent receives a reward $R = -(C - A)^2 + \epsilon_r$, where the exogenous noise ϵ_r is sampled from $\mathcal{N}(0, \sigma_r)$, as well as a feedback vector F which is a function of C, A and ϵ_r. More details about this problem as well as variants are presented in Appendix B.1.

[4] Note that from this point of view, actions at future time-step are effectively 'chance' from the point of view of computing credit for action A_t

For this problem, the optimal policy is to choose $A = C$, resulting in average reward of 0. However, the reward signal R is corrupted by the exogenous noise ϵ_r, uncorrelated to the action. The higher the standard deviation, the more difficult proper credit assignment becomes, as high rewards are more likely due to a high value of ϵ_r than an appropriate choice of action. On the other hand, the feedback F contains information about C, A and ϵ_r. If the agent can extract information Φ from F in order to capture information about ϵ_r and use it to compute a hindsight value function, the effect of the perturbation ϵ_r may be removed from the advantage estimate, resulting in a significantly lower variance estimator. However, if the agent blindly uses F to compute the hindsight value information, information about the action will 'leak' into the hindsight value, leading to an advantage estimate of 0 and no learning.

We investigate the proposed algorithm with $N = 10$. As can be seen on Fig. 2, increasing the variance of the exogenous noise leads to dramatic decrease of performance for the vanilla PG estimator without the hindsight baseline; in contrast, the CCA-PG estimator is generally unaffected by the exogenous noise. For very low level of exogenous noise however, CCA-PG suffers from a decrease in performance. This is due to the agent computing a hindsight statistic Φ which is not perfectly independent from A, leading to bias in the policy gradient update. To demonstrate this effect, and evaluate the importance of the independence constraint on performance, we run an ablation where we test lower values of the weight λ_{IM} of the independence maximization loss (leading to a larger mutual information between Φ and A) and indeed observed that the performance is dramatically degraded, as seen in Fig. 2.

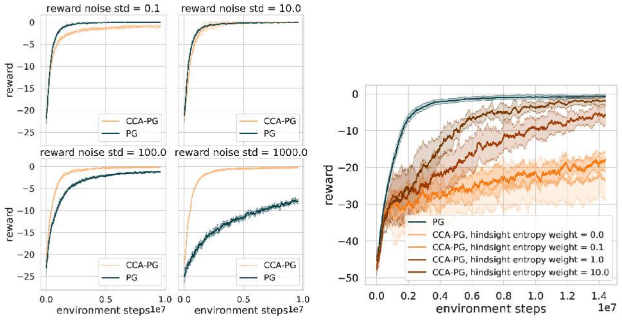

Figure 2: Top: Comparison of CCA-PG and PG in contextual bandits with feedback, for various levels of reward noise σ_r. Results are averaged over 6 independent runs with standard deviation represented by a shaded area. **Bottom:** Performance of CCA-PG on the bandit task, for different values of λ_{IM}. Properly enforcing the independence constraint prevents the degradation of performance.

4.2. Key-to-Door environments

Task Description. We investigate new versions of the Key-To-Door family of environments, initially proposed by Hung et al. (2019), as a testbed of tasks where credit assignment is hard and is necessary for success. In this partially observable grid-world environment (cf. Fig. 7 in the appendix), the agent has to pick up a key in the first room, for which it has *no immediate reward*. In the second room, the agent can pick up 10 apples, that each give immediate rewards. In the final room, the agent may open a door (only if it is carrying a key), and receive a small reward for doing so. In this task, a single action (i.e picking up the key) has a direct impact on the reward it receives in the final room, however this signal is hard to detect as the episode return is largely driven by its performance in the second room (i.e picking up apples).

We now consider two instances of the Key-To-Door family that illustrate the difficulty of credit assignment in the presence of extrinsic variance. In the Low-Variance-Key-To-Door environment, each apple is worth a reward of 1 and opening the final door also gets a reward of 1. Thus, an agent that solves the apple phase perfectly sees very little variance in its episode return and the learning signal for picking up the key and opening the door is relatively strong.

High-Variance-Key-To-Door keeps the overall structure of the Key-To-Door task. The door keeps giving a deterministic reward of 1 when the key was grabbed but now the reward for each apple is randomly sampled to be either 1 or 10, and fixed within the episode. In this setting, even an agent that is skilled at picking up apples sees a large variance in episode returns, and thus the learning signal for picking up the key and opening the door is comparatively weaker. Appendix B.2.1 has some additional discussion illustrating the difficulty of learning in such a setting.

Results We test CCA-PG on these environments, and compare it against Actor-Critic (Williams, 1992), as well as State-conditional HCA and Return-conditional HCA (Harutyunyan et al., 2019) as baselines. An analysis of the relation between HCA and CCA is described in Appendix C. We test using both a backward-LSTM (referred to as CCA-PG RNN) or an attention model (referred to as CCA-PG Attn) for the hindsight function. Details for experimental setup are provided in Appendix B.2.2.

We evaluate agents both on their ability to maximize total reward, as well as to solve the specific credit assignment problem of picking up the key and opening the door. Fig. 3 compares CCA-PG with the baselines on the High-Variance-Key-To-Door task. Both CCA-PG architectures outperform the baselines in terms of total reward, as well as probability of picking up the key and opening the door.

This experiment highlights the capacity of CCA-PG to learn and incorporate trajectory-specific external factors into its baseline, resulting in lower variance estimators. Despite being a difficult task for credit assignment, CCA-PG is capable of solving it quickly and consistently. On the other

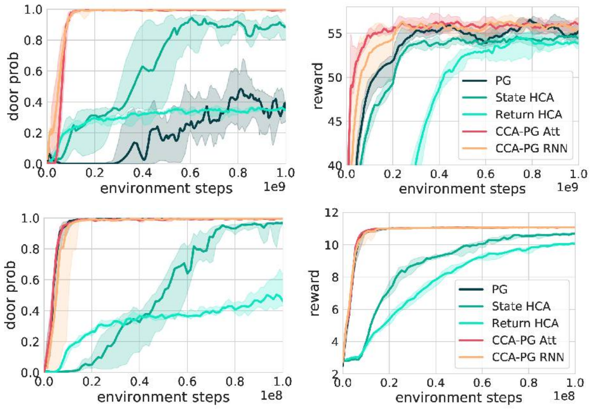

Figure 3: Probability of opening the door and total reward obtained on the **High-Variance-Key-To-Door** task (top two) and the **Low-Variance-Key-To-Door** task (bottom two).

hand, vanilla actor-critic is greatly impacted by this external variance, and needs around 3.10^9 environment steps to have an 80% probability of opening the door. CCA-PG also outperforms State- and Return- Conditional HCA, which do use hindsight information but in a more limited way than CCA-PG.

On the Low-Variance-Key-To-Door task (Fig. 3), due to the lack of extrinsic variance, standard actor-critic is able to perfectly solve the environment. However, it is interesting to note that CCA-PG still matches this perfect performance. On the other hand, the other hindsight methods struggle with both door-opening and apple-gathering. This might be explained by the fact that both these techniques do not guarantee lower variance, and rely strongly on their learned hindsight classifiers for their policy gradient estimators, which can be harmful when these quantities are not perfectly learned. See Appendix B.2.3 for additional experiments and ablations on these environments.

These experiments demonstrate that CCA-PG is capable of efficiently leveraging hindsight information to mitigate the challenge of external variance and learn strong policies that outperform baselines. At the same time, it suffers no drop in performance when used in cases where external variance is minimal.

4.3. Task Interleaving

Motivation. In the real world, human activity can be seen as solving a large number of loosely related problems. At an abstract level, one could see this lifelong learning process as solving problems not in a sequential, but an interleaved fashion instead. These problems are not solved sequentially, as one may temporarily engage with a problem and only continue engaging with it or receive feedback from its earlier actions significantly later. The structure of this interleaving will also typically vary over time.

To better understand the effects of interleaving on agent learning, we introduce a new class of environments capturing the structural properties mentioned above. In contrast to most work on multi-task learning, we do not assume a clear delineation between subtasks, nor focus on skill retention. The agent will encounter multiple tasks in a single episode in an interleaved fashion (switching between tasks will occur before a task gets completed), and will have to detect the implicitly boundaries between them.

Task Description. This task consists of pairs of query-answer rooms with different visual contexts that each indicates a different subtask. In the query room, the agent gets to pick between two colored boxes (out of 10 possible colors). Later, in the answer room, the agents gets to observe which of the two boxes was rewarding in the first room, and receives a reward if it picked the correct box (there is always exactly one rewarding color in the query room). The mapping of colors to whether it is rewarding or not is specific to each subtask and fixed across training. Each subtask would be relatively easy to solve if encountered in an isolated fashion. However, each episode is composed of *randomly sampled subtasks* and color pairs within those subtasks. Furthermore, query rooms and answer rooms of the sampled subtasks are presented in a random (interleaved) order which differs from one episode to another. Each episode are 140 steps long and it takes at least 9 steps for the agent to reach one colored square from its initial position. A visual example of what an episode looks like can be seen in Fig. 4.

There are six tasks, each classified as 'easy' or 'hard'; easy tasks have high reward signals (i.e. easier for agents to pick up on), while hard tasks have low rewards. In the 2 tasks setup (resp. 4 tasks and 6 tasks), there is one (resp. two and two) 'easy' and one (resp. two and four) 'hard' task. More details about the experimental setup can be found in B.3.

In addition to the total reward, we record the probability of picking up the correct square for the easy and hard tasks separately. Performance in the hard tasks will indicate ability to do fine-grained credit assignment.

Results. While CCA-PG is able to perfectly solve both the 'easy' and 'hard' tasks in the three setups in less than 5.10^8 environment steps (Fig. 5), actor-critic is only capable to solve the 'easy' tasks for which the associated rewards are large. Even after 2.10^9 environment steps, actor-critic is still greatly impacted by the variance and remains incapable of solving 'hard' tasks in any of the three settings. CCA-PG also outperforms actor-critic in terms of the total reward obtained in each setting. State-conditional and Return-conditional HCA were also evaluated on this task but results are not reported as almost no learning was taking place on the 'hard' tasks. More results along with an ablation study can be found in Appendix B.3.

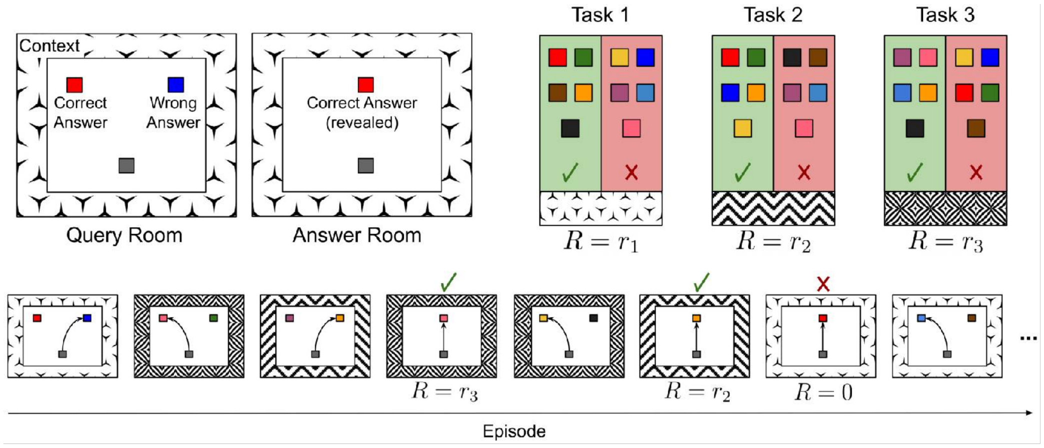

Figure 4: Task Interleaving Description. Top left: Delayed feedback contextual bandit problem. Given a context shown as a surrounding visual pattern, the agent has to decide to pick up one of the two colored squares where only one will be rewarding. The agent is later teleported to the second room where it is provided with the reward associated with its previous choice and a visual cue about which colored square it should have picked up. **Top right:** Different tasks with each a different color mapping, visual context and associated reward. **Bottom:** Example of a generated episode, composed of randomly sampled tasks and color pairs.

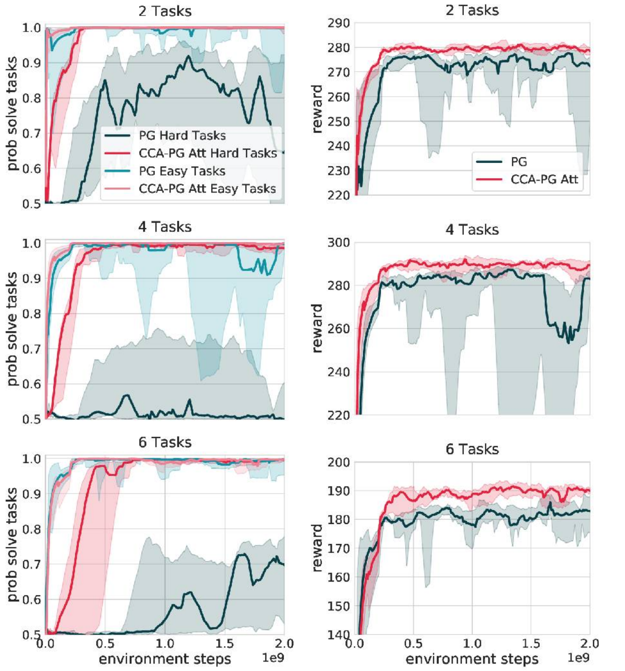

Figure 5: Probability of solving 'easy' and 'hard' tasks (left) and total reward (right) obtained for the **Multi Task Interleaving.** Left plots: Median over 10 seeds after doing a mean over the performances in 'easy' or 'hard' tasks.

Through efficient use of hindsight, CCA-PG is able to take into account trajectory-specific factors such as the kinds of rooms encountered in the episode and their associated rewards.

In the case of the Multi-Task Interleaving environment, an informative hindsight function would capture the reward for different contexts and exposes as Φ_t all rewards obtained in the episode except those associated with the current context. This experiment again highlights the capacity of CCA-PG to solve hard credit assignment problems in a context where the return is affected by multiple distractors, while PG remains highly sensitive to them.

5. Related work

This paper builds on work from Buesing et al. (2019) which shows how causal models and real data can be combined to generate counterfactual trajectories and perform off-policy evaluation for RL. Their results however require an explicit model of the environment. In contrast, our work proposes a model-free approach, and focuses on policy improvement. Oberst & Sontag (2019) also investigate counterfactuals in reinforcement learning, point out the issue of non-identifiability of the correct SCM, and suggest a sufficient condition for identifiability; we discuss this issue in appendix G. Closely related to our work is Hindsight Credit Assignment, a concurrent approach from Harutyunyan et al. (2019). In this paper, the authors also investigate value functions and critics that depend on future information. However, the information the estimators depend on is fixed (future state or return) instead of being an arbitrary functions of the trajectory. Our FC estimators generalizes both the HCA and CCA estimators while CCA further characterizes which statistics of the future provide a useful estimator. Relations between HCA, CCA and FC are discussed in appendix C. The HCA approach is further extended by Young (2019), and Zhang et al. (2019) who minimize a surrogate for the variance of the estimator, but that surrogate cannot be guaranteed to actually lower the variance. Similarly to state-HCA, it treats each reward separately instead of taking a trajectory-centric view as CCA. Guez et al. (2019) also investigate future-conditional value functions. Similarly to us, they learn statistics of the future Φ from which returns can be accurately predicted, and show that doing so leads to learning better representations (but use regular policy gradient estimators otherwise). Instead of enforcing a information-theoretic constraint, they bottleneck information through the size of the encoding Φ. In domain adaptation (Ganin et al., 2016; Tzeng et al., 2017), robustness to the training domain can be achieved by constraining the agent representation not to be able to discriminate between source and target domains, a mechanism similar to the one constraining hindsight features not being able to discriminate the agent's actions. Also closely related to our paper, Bica et al. (2020) also leverages a similar mechanism to compute counterfactuals, for a different purpose than ours (computing treatment effects vs. policy improvement operators).

Both Andrychowicz et al. (2017) and Rauber et al. (2017) leverage the idea of using hindsight information to learn goal-conditioned policies. Hung et al. (2019) leverages attention-based systems and episode memory to perform long term credit assignment; however, their estimator will in general be biased. Ferret et al. (2019) looks at the question of transfer learning in RL and leverages transformers to derive a heuristic to perform reward shaping. Arjona-Medina et al. (2019) also addresses the problem of long-term credit assignment by redistributing delayed rewards earlier in the episode but their approach still fundamentally uses time as a proxy for credit.

Previous research also leverages the fact that baselines can include information unknown to the agent at time t (but potentially revealed in hindsight) but not affected by action A_t: for instance, when using independent multi-dimensional actions, the baseline for one dimension of the action vector can include the actions in other dimensions (Wu et al., 2018); or when the dynamic of the environment is partially driven by an exogenous and stochastic factor, independent of the agent's actions, which can be included in the baseline (Mao et al., 2018). Similarly, in multi-agent environments, actions of other agents at the same time step (Foerster et al., 2018) can be used; and so can the full state of the simulator when learning control from pixels (Andrychowicz et al., 2020), or the use of opponent observations in Starcraft II (Vinyals et al., 2019). Note however that all of these require privileged information, both in the form of feeding information to the baseline inaccessible to the agent, and in knowing that this information is independent from the agent's action A_t and therefore won't bias the baseline. Our approach seeks to replicate a similar effect, but in a more general fashion and from an agent-centric point of view, where the agent *learns itself* which information from the future can be used to improve its baseline at time t.

6. Conclusion

In this paper we have considered the problem of credit assignment in RL. Building on insights from causality theory and structural causal models, we have investigated the concept of future-conditional value functions. Contrary to common practice these allow baselines and critics to condition on future events thus separating the influence of an agent's actions on future rewards from the effects of other random events thus reducing the variance of policy gradient estimators. A key difficulty lies in the fact that unbiasedness relies on accurate estimation and minimization of mutual information. Learning inaccurate hindsight classifiers will result in miscalibrated estimation of luck, leading to bias in learning. Future research will investigate how to scale these algorithms to more complex environments, and the benefits of the more general FC-PG and all-actions estimators.

References

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, O. P., and Zaremba, W. Hindsight experience replay. In *Advances in neural information processing systems*, pp. 5048–5058, 2017.

Andrychowicz, O. M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Arjona-Medina, J. A., Gillhofer, M., Widrich, M., Unterthiner, T., Brandstetter, J., and Hochreiter, S. Rudder: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, pp. 13544–13555, 2019.

Bica, I., Alaa, A. M., Jordon, J., and van der Schaar, M. Estimating counterfactual treatment outcomes over time through adversarially balanced representations. *arXiv preprint arXiv:2002.04083*, 2020.

Buesing, L., Weber, T., Zwols, Y., Racaniere, S., Guez, A., Lespiau, J.-B., and Heess, N. Woulda, coulda, shoulda: Counterfactually-guided policy search. *2019 International Conference for Learning Representations (ICLR)*, 2019.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Gated feedback recurrent neural networks. In *International conference on machine learning*, pp. 2067–2075, 2015.

Ferret, J., Marinier, R., Geist, M., and Pietquin, O. Credit assignment as a proxy for transfer in reinforcement learning. *arXiv preprint arXiv:1907.08027*, 2019.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Glasserman, P. and Yao, D. D. Some guidelines and guarantees for common random numbers. *Management Science*, 38(6):884–908, 1992.

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

Guez, A., Viola, F., Weber, T., Buesing, L., Kapturowski, S., Precup, D., Silver, D., and Heess, N. Value-driven hindsight modelling. *https://openreview.net/forum?id=rJxBa1HFvS*, 2019.

Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Hamrick, J. B. Analogues of mental simulation and imagination in deep learning. *Current Opinion in Behavioral Sciences*, 29:8–16, 2019.

Harutyunyan, A., Dabney, W., Mesnard, T., Azar, M. G., Piot, B., Heess, N., van Hasselt, H. P., Wayne, G., Singh, S., Precup, D., et al. Hindsight credit assignment. In *Advances in Neural Information Processing Systems*, pp. 12467–12476, 2019.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, pp. 2944–2952, 2015.

Hinton, G., Srivastava, N., and Swersky, K. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8), 2012.

Hung, C.-C., Lillicrap, T., Abramson, J., Wu, Y., Mirza, M., Carnevale, F., Ahuja, A., and Wayne, G. Optimizing agent behavior over long time scales by transporting value. *Nature communications*, 10(1):1–12, 2019.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Mao, H., Venkatakrishnan, S. B., Schwarzkopf, M., and Alizadeh, M. Variance reduction for reinforcement learning in input-driven environments. In *International Conference on Learning Representations*, 2018.

Minsky, M. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

Oberst, M. and Sontag, D. Counterfactual off-policy evaluation with gumbel-max structural causal models. *arXiv preprint arXiv:1905.05824*, 2019.

Parisotto, E., Song, H. F., Rae, J. W., Pascanu, R., Gulcehre, C., Jayakumar, S. M., Jaderberg, M., Kaufman, R. L., Clark, A., Noury, S., et al. Stabilizing transformers for reinforcement learning. *arXiv preprint arXiv:1910.06764*, 2019.

Pearl, J. *Causality*. Cambridge university press, 2009a.

Pearl, J. Causality: Models, reasoning, and inference. 2009b.

Rauber, P., Ummadisingu, A., Mutz, F., and Schmidhuber, J. Hindsight policy gradients. *arXiv preprint arXiv:1711.06006*, 2017.

Rezende, D. J. and Viola, F. Taming VAEs. *arXiv preprint arXiv:1810.00597*, 2018.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *arXiv preprint arXiv:1911.08265*, 2019.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7167–7176, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft II using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Weber, T., Heess, N., Buesing, L., and Silver, D. Credit assignment techniques in stochastic computation graphs. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2650–2660, 2019.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Wu, C., Rajeswaran, A., Duan, Y., Kumar, V., Bayen, A. M., Kakade, S., Mordatch, I., and Abbeel, P. Variance reduction for policy gradient with action-dependent factorized baselines. *2018 International Conference for Learning Representations (ICLR)*, 2018.

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

Zhang, J. Designing optimal dynamic treatment regimes: A causal reinforcement learning approach. In *International Conference on Machine Learning*, pp. 11012–11022. PMLR, 2020.

Zhang, P., Zhao, L., Liu, G., Bian, J., Huang, M., Qin, T., and Tie-Yan, L. Independence-aware advantage estimation. *https://openreview.net/forum?id=B1eP504YDr*, 2019.

Appendix

A. Algorithmic and implementation details

A.1. Constrained optimization

Corollary 1 requires an independence assumption between A_t and Φ_t, conditional on X_t. We can therefore cast the problem of learning Φ_t as a constrained optimization problem, where the loss $\mathcal{L}_{\text{hs}}$ measures how predictive of the return Φ_t is, and the constraint enforces a maximum (tolerance) value of β_{IM} for the independence maximization loss $\mathcal{L}_{\text{IM}}$ (exact independence is obtained by $\beta_{\text{IM}} = 0$, but this is hard to achieve exactly in practice).

The resulting optimization problem for finding an appropriate counterfactual baseline is given by:

$$\min_{\theta} \mathbb{E}\left[\mathcal{L}_{\text{hs}}\right] \quad \text{subject to: } \forall t \;\; \mathcal{L}_{\text{IM}}(X_t) \leq \beta_{\text{IM}} \tag{7}$$

The resulting hindsight baseline can then be used in the policy gradient estimate. There are two problems remaining to solve. First, the form of the (IM) loss used requires knowing the exact hindsight probability $\mathbb{P}(A_t|X_t, \Phi_t)$. As explained in the main text, we replace it by the classifier h, tracking the optimal classifier by stochastically minimizing the supervised loss (optimizing it only with respect to the parameters of the hindsight classifier). Second, we relax the constraint using a Lagrangian method (the Lagrangian parameter can either be set as a hyperparameter, or optimized using an algorithm like GECO (Rezende & Viola, 2018)).

A.2. Parameter updates

The corresponding parameter updates are as follows:

For each trajectory $(X_t, A_t, R_t)_{t\geq 0}$, compute the parameter updates :

- $\Delta\theta_{\text{fs}} = -\lambda_{\text{PG}} \sum_t \gamma^t \nabla_{\theta_{\text{fs}}} \log \pi(A_t|X_t)(G_t - V(X_t, \Phi_t)) + \lambda_{\text{H}} \sum_t \nabla_{\theta_{\text{fs}}} \mathcal{L}_{\text{H}}(t) + \lambda_{\text{hs}} \sum_t \nabla_{\theta_{\text{fs}}} \mathcal{L}_{\text{hs}}(t)$ where $\mathcal{L}_{\text{H}}(t) = -\sum_a \pi(a|X_t) \log \pi(a|X_t)$ is an entropy bonus.
- $\Delta\theta_{\text{hs}} = \lambda_{\text{hs}}(t) \sum_t \nabla_{\theta_{\text{hs}}} \mathcal{L}_{\text{hs}} + \lambda_{\text{IM}} \sum_t \nabla_{\theta_{\text{hs}}} (\mathcal{L}_{\text{IM}}(t) - \beta \mathbb{H}[A_t|X_t])$
- $\Delta\omega = \sum_t \nabla_{\omega} \mathcal{L}_{\text{sup}}(t)$
- $\Delta\lambda_{\text{IM}} = -\lambda_{\text{IM}} \sum_t (\mathcal{L}_{\text{IM}}(t) - \beta \mathbb{H}[A_t|X_t])$ (when using GECO)

A.3. Design choices

Here we detail practical choices for two aspects of the general CCA algorithm. These concern a) the form of the hindsight function, b) the form of the independence maximization constraint.

Choice of the hindsight function φ

In principle, this function can take any form: in practice, we investigated two architectures. The first is a backward RNN, where $(\Phi_t, B_t) = \text{RNN}(X_t, B_{t+1})$, where B_t is the state of the backward RNN. Backward RNNs are justified in that they can extract information from arbitrary length sequences, and allow making the statistics Φ_t a function of the entire trajectory. They also have the inductive bias of focusing more on near-future observations. The second is a transformer (Vaswani et al., 2017; Parisotto et al., 2019). Alternative networks could be used, such as attention-based networks (Hung et al., 2019) or RIMs (Goyal et al., 2019).

Independence maximization constraint $\mathcal{L}_{\text{IM}}$

We investigated two IM losses. The first is the conditional mutual information $\mathbb{I}(A_t; \Phi_t|X_t) = \mathbb{E}_{\Phi_t|X_t}[\mathbb{H}[A_t|X_t] - \mathbb{H}[A_t|X_t, \Phi_t]]$, where $\mathbb{H}[A|B]$ denotes the conditional entropy $\mathbb{H}[A|B] = -\sum_a P(A = a|B) \log P(A = a|B)$. The expectation can be stochastically approximated by the trajectory sample value $\mathbb{H}(A_t|X_t) - \mathbb{H}(A_t|X_t, \Phi_t)$. The first term is simply the entropy of the policy $-\sum_a \pi(a|X_t) \log \pi(a|X_t)$. The second term is estimated using the h network. The second we investigated is the Kullback-Leibler divergence, $\mathbb{KL}(\pi(A_t|X_t)||\mathbb{P}(A_t|X_t,\Phi_t)) = \sum_a \pi(a|X_t)\log\pi(a|X_t) - \sum_a \pi(a|X_t)\log\mathbb{P}(a|X_t,\Phi_t)$. Again, we approximate the second term using h. We did not see significant differences between the two, with the KL slightly outperforming the mutual information.

B. Additional Experimental Details

B.1. Bandits

B.1.1. Environment

Our bandit with feedback environment is defined by two positive integers (N, K), a noise level $\sigma_r > 0$ and three arbitrary matrices U, V, W, where $U, V \in \mathbb{R}^{K\times N}$ and $W \in \mathbb{R}^K$. For each replication of the experiment (i.e. each seed), these matrices are sampled from a standard Gaussian distribution and kept constant throughout all episodes. For each episode (of length 1, since this is a bandit problem tackled without meta-learning), we sample a context $-N \leq C \leq N$. Given C, an agent chooses an action $-N \leq A \leq N$. The agent then receives a reward $R = -(C-A)^2 + \epsilon_r$, where ϵ_r is sampled from $\mathcal{N}(0,\sigma_r)$. The agent additionally receives a K-dimensional feedback vector $F = U_C + V_A + W\epsilon_r$, where U_C (resp. V_A) denotes the C$^{\text{th}}$ (resp. A$^{\text{th}}$) column of U (resp. V).

The choices above were made without any particular intent: we would expect the intuitions to generalize for other noise distributions and feedback functions. In section B.1.3, we investigate a decentralized multiagent variant of this problem where the exogenous noise actually corresponds to other players' actions.

B.1.2. Architecture

For the bandit problems, the agent architecture is as follows:

- The hindsight feature Φ is computed by a backward RNN. We tried multiple cores for the RNN: GRU (Chung et al., 2015) with 32 hidden units, a recurrent adder ($h_t = h_{t-1} + \text{MLP}(x_t)$, where the MLP has two layers of 32 units), or an exponential averager ($h_t = \lambda h_{t-1} + (1-\lambda)\text{MLP}(x_t)$).
- The hindsight classifier h_ω is a simple MLP with two hidden layers with 32 units each.
- The policy and value functions are computed as the output of a simple linear layer with concatenated observation and feedback as input.
- All weights are jointly trained with Adam (Kingma & Ba, 2014).
- Hyperparameters are chosen as follows: learning rate $4e\text{-}4$, entropy loss $4e\text{-}3$, independence maximization tolerance $\beta_{\text{IM}} = 0.1$, $\lambda_{\text{sup}} = \lambda_{\text{hs}} = 1$, λ_{IM} is set through Lagrangian optimization (with GECO).

B.1.3. Additional results

Multi-agent Bandit Problem: In the multi-agent version, the environment is composed of M replicas of the bandit with feedback task. Each agent $i = 1,\ldots,M$ interacts with its own version of the environment, but feedbacks and rewards are coupled across agents. The multi-agent bandit is obtained by modifying the single agent version as follows:

- The contexts C^i are sampled i.i.d. from $\{-N,\ldots,N\}$. C and A now denote the concatenation of all agents' contexts and actions.
- The feedback tensor is (M,K) dimensional, and is computed as $W_c\mathbf{1}(C) + W_a\mathbf{1}(A) + \epsilon_f$; where the W are now three dimensional tensors. Effectively, the feedback for agent i depends on the context and actions of all other agents.
- The terminal joint reward is $\sum_i -(C^i - A^i)^2$ for all agents.

The multi-agent version does not require the exogenous noise ϵ_r, as other agents play the role of exogenous noise; it is a minimal implementation of the example found in section 2.3.

We report results from the multi-agent version of the environment in Fig. 6. As the number of interacting agents increases, the effective variance of the vanilla PG estimator increases as well, and the performance of each agent decreases. In contrast, CCA-PG agents learn faster and reach higher performance (though they never learn the optimal policy).

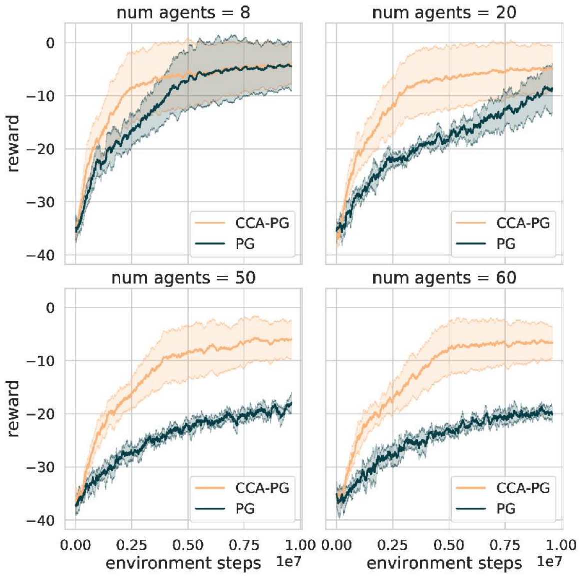

Figure 6: Multi-agent versions of the bandit problem. CCA-PG agents outperform vanilla PG ones.

B.2. Key to Door Tasks

B.2.1. Environment details

Observations returned by the key-to-door family of environments for each of the three phases can be visualized in Fig. 7.

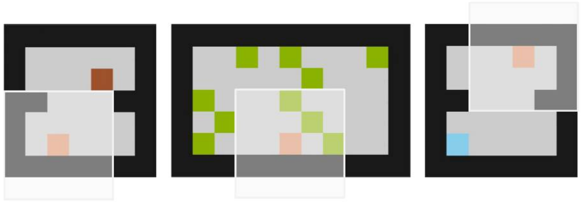

Figure 7: Key-To-Door environments visual. The agent is represented by the beige pixel, key by brown, apples by green, and the final door by blue. The agent has a partial field of view, highlighted in white.

		Lucky (high apple reward)	Unlucky (low apple reward)
Hindsight Advantage Estimate	Skillful (Got key + Door)	1	1
	Unskillful (Did not get key or door)	0	0
Forward Advantage Estimate	Skillful (Got key + Door)	46	-44
	Unskillful (Did not get key or door)	45	-45

Table 1: The advantage estimate of the action of picking up a key in High-Variance-Key-To-Door, as computed by an agent that always picks up every apple, and never picks up the key or the door. We see that an advantage estimate learned using hindsight clearly differentiates between the skillful and unskillful actions; whereas for an advantage estimate learned without using hindsight, this difference is dominated by the extrinsic randomness.

To motivate our approach, Table 1 shows the advantage estimates for either picking up the key or not on High-Variance-Key-To-Door, for an agent that has a perfect apple-phase policy, but never picks up the key or door. Since there are 10 apples which can be worth 1 or 10, the return will be either 10 or 100. Thus the forward baseline in the key phase, i.e. before it has seen how much an apple is worth in the current episode, will be 55. As seen in Table 1, the difference in advantage estimates due to 'luck' is far larger than the difference in advantage estimates due to 'skill' when not using hindsight. This makes learning difficult and leads to the policy never learning to start picking up the key or opening the door. However, when we use a hindsight-conditioned baseline, we are able to learn a Φ (such as the value of a single apple in the current episode) that is completely independent from the actions taken by the agent, but which can provide a perfect hindsight-conditioned baseline of either 10 or 100.

B.2.2. Architecture

The agent architecture is as follows:

- The observations are first fed to 2-layer CNN with $(16, 32)$ output channels, kernel shapes of $(3, 3)$ and strides of $(1, 1)$. The output of the CNN is flattened and fed to a linear layer of size 128.
- The agent state is computed by a forward LSTM with a state size of 128. The input to the LSTM is the output of the previous linear layer, concatenated with the reward at the previous timestep.
- The hindsight feature Φ is computed either by a backward LSTM (i.e CCA-PG RNN) with a state size of 128 or by an attention mechanism (Vaswani et al., 2017) (i.e CCA-PG Att) with value and key sizes of 64, 1 transformer block with 2 attention heads, a 1 hidden layer MLP of size 1024, an output size of 128 and a rate of dropout of 0.1. The input provided is the concatenation of the output of the forward LSTM and the reward at the previous timestep.
- The policy is computed as the output of a single-layer MLP with 64 units where the output of the forward LSTM is provided as input.
- The forward baseline is computed as the output of a 3-layer MLP of 128 units each where the output of the forward LSTM is provided as input.
- The hindsight baseline is computed as the sum of the forward baseline and a hindsight residual baseline; the hindsight residual baseline is the output of a 3-layer MLP of 128 units each where the concatenation of the output of the forward LSTM and the hindsight feature Φ is provided as input. It is trained to learn the residual between the return and the forward baseline.
- For CCA, the hindsight classifier h_ω is computed as the concatenation of the output of an MLP, with four hidden layers with 256 units each where the concatenation of the output of the forward LSTM and the hindsight feature Φ is provided as input, and the log of the policy outputs.
- For State HCA, the hindsight classifier h_ω is computed as the output of an MLP, with four hidden layers with 256 units each, where the concatenation of the outputs of the forward LSTM at two given time steps is provided as input.
- For Return HCA, the hindsight classifier h_ω is computed as the output of an MLP, with four hidden layers with 256 units each, where the concatenation of the output of the forward LSTM and the return is provided as input.
- All weights are jointly trained with RMSprop (Hinton et al., 2012) with epsilon $1e\text{-}4$, momentum 0 and decay 0.99.

For High-Variance-Key-To-Door, the optimal hyperparameters found for each algorithm can be found in Table 2.

For Key-To-Door, the optimal hyperparameters found for each algorithm can be found in Table 3.

The agents are trained on full-episode trajectories, using a discount factor of 0.99.

B.2.3. Additional results

As shown in Fig. 8, in the case of vanilla policy gradient, the baseline loss increases at first. As the reward associated with apples varies from one episode to another, getting more apples also means increasing the forward baseline loss. On the

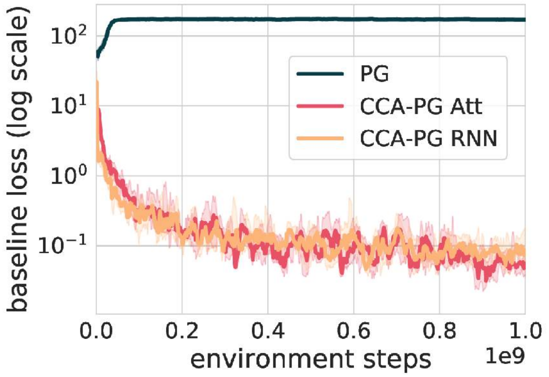

Figure 8: Baseline loss for vanilla PG versus hindsight baseline loss for CCA in **High-Variance-Key-To-Door**.

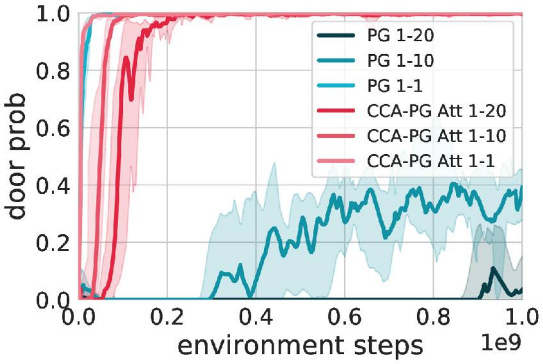

Figure 9: Impact of variance over credit assignment performances. Probability of picking up the key and opening the door as a function of the variance level induced by the apple reward discrepancy between episodes.

other hand, as CCA is able to take into account trajectory specific exogenous factors, the hindsight baseline loss can nicely decrease as learning takes place.

Fig. 9 shows the impact of the variance level induced by the apple reward discrepancy between episodes on the probability of picking up the key and opening the door. Thanks to the use of hindsight in its value function, CCA-PG is almost not impacted by this whereas vanilla PG sees its performances drop dramatically as variance increases.

Fig. 10 shows a qualitative analysis of the attention weights learned by CCA-PG Att on the High-Variance-Key-To-Door task. For this experiment, we used only a single attention head for easier interpretation of the hindsight function, and show both a heatmap of the attention weights over the entire episode, and a histogram of attention weights at the step where the agent picks up the key. As expected, the most attention is paid to timesteps just after the agent picks up an apple - since these are the points at which the apple reward is provided to the Φ computation. In particular, very little attention is paid to the timestep where the agent opens the door. These insights further show that the hindsight function learned is highly predictive of the episode return, while not having mutual information with the action taken by the agent, thus ensuring an unbiased policy gradient estimator.

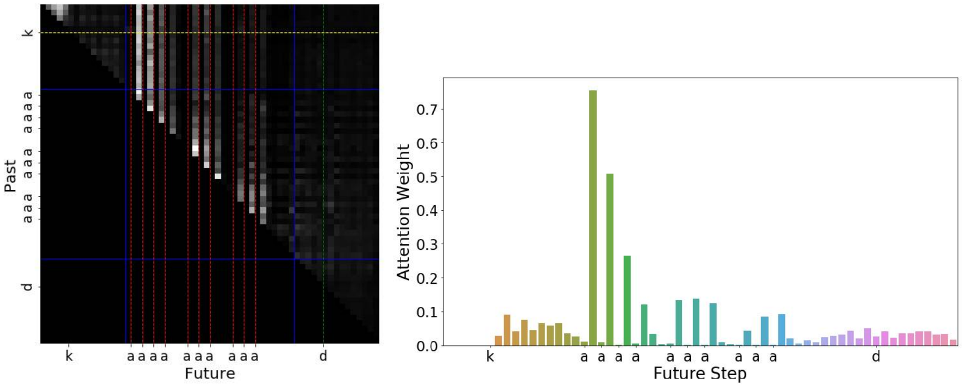

Figure 10: Visualization of attention weights on the High-Variance-Key-To-Door task. **Top:** a 2-dimensional heatmap showing how the hindsight function at each step attends to each step in the future. Red lines indicate the timesteps at which apples are picked up (marked as ‘a’); green indicates the door (marked as ‘d’); yellow indicates the key (marked as ‘k’). **Bottom:** A bar plot of attention over future timesteps, computed at the step where the agent is just about to pick up the key.

	CCA Att	CCA RNN	PG	State HCA	Return HCA
Policy cost	1	1	1	1	1
Entropy cost	5e-3	5e-3	5e-3	5e-3	5e-3
Forward baseline cost	5e-2	5e-2	5e-2	5e-2	5e-2
Hindsight residual baseline cost	5e-2	5e-2	—	———	———
Hindsight classifier cost	1e-2	1e-2	—	1e-2	1e-2
Action independence cost	1e2	1e2	—	———	———
Learning rate	5e-4	5e-4	1e-4	5e-4	1e-3

Table 2: High-Variance-Key-To-Door hyperparameters

	CCA Att	CCA RNN	PG	State HCA	Return HCA
Policy cost	1	1	1	1	1
Entropy cost	5e-3	5e-3	5e-3	5e-3	5e-3
Forward baseline cost	5e-2	5e-2	5e-2	5e-2	5e-2
Hindsight residual baseline cost	5e-2	5e-2	—	———	———
Hindsight classifier cost	1e-2	1e-2	—	1e-2	1e-2
Action independence cost	1e2	1e2	—	———	———
Learning rate	5e-4	5e-4	5e-4	5e-4	5e-4

Table 3: Key-To-Door hyperparameters

B.3. Task Interleaving

B.3.1. Environment details

For each task, a random, but fixed through training, set of 5 out of 10 colored squares are leading to a positive reward. Furthermore, a small reward of 0.5 is provided to the agent when it picks up any colored square. As mentioned previously, each episode are 140 steps long and it takes at least 9 steps for the agent to reach one colored square from its initial position.

The 6 tasks we consider (numbered #1 to #6) are respectively associated with a reward of 80, 4, 100, 6, 2 and 10. Tasks #2, #4, #5 and #6 are referred to as 'hard' while tasks #1 and #3 as 'easy' because of their large associated rewards. The settings 2, 4 and 6-task are respectively considering tasks 1-2, 1-4 and 1-6.

B.3.2. Architecture

We use the same architecture setup as reported in Appendix B.2.2. The agents are also trained on full-episode trajectories, using a discount factor of 0.99.

For Task Interleaving, the optimal hyperparameters found for each algorithm can be found in Table 4.

B.3.3. Additional results

Fig. 11 shows that CCA is able to solve all 6 tasks quickly despite the variance induced by the exogenous factors. Vanilla PG on the other hand despite solving the 'easy' tasks 1 and 3 for which the agent receives big rewards, it is incapable of reliably solve the 4 remaining tasks for which the associated reward is smaller. This helps unpacking Fig. 5.

B.3.4. Ablation Study

Fig.12 shows the impact of the number of back-propagation through time steps performed into the backward RNN of the hindsight function while performing full rollouts. This shows that learning in 'hard' tasks, i.e. where hindsight is crucial for performances, is not much impacted by the number of back-propagation steps performed into the backward RNN. This is great news as this indicates that learning in challenging credit assignment tasks still works when the hindsight function sees the whole future but can only backprop through a limited window.

Fig.13 shows how performances of CCA-RNN are impacted by the unroll length. As expected, the less it is able to look into the future, the harder it becomes to solve hard credit assignment tasks as it is limited in its capacity to take into account exogenous effects.

The two previous results are promising since CCA seems to only require to have access to as many steps into the future as possible while not needing to do back-propagation through the full sequence.

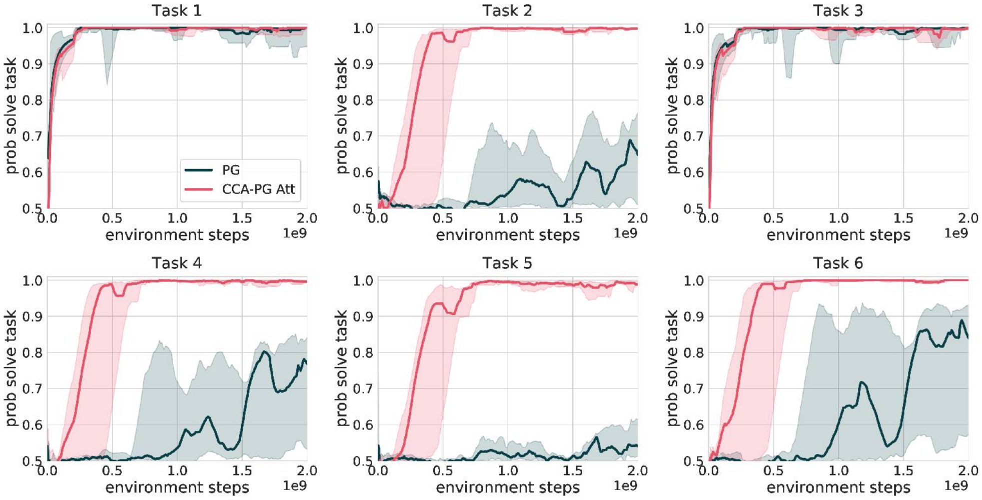

Figure 11: Probability of solving each task in the 6-task setup for **Task Interleaving**.

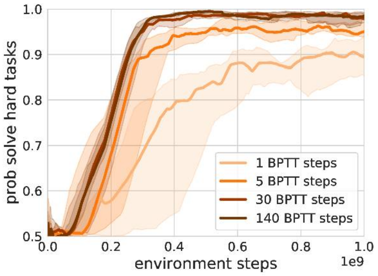

Figure 12: Impact of the number of back-propagation through time steps performed into the hindsight function for CCA-RNN. Probability of solving the 'hard' tasks in the 6-task setup of **Task Interleaving**.

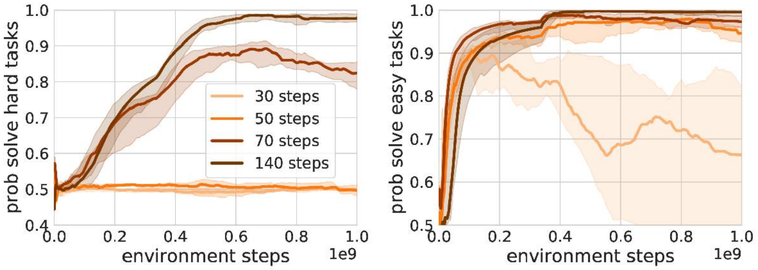

Figure 13: Impact of the unroll length for CCA-RNN. Probability of solving the 'hard' and 'easy' tasks in the 6-task setup of **Task Interleaving**.

	CCA Att	CCA RNN	PG
Policy cost	1	1	1
Entropy cost	5e-2	5e-2	5e-2
Forward baseline cost	1e-2	5e-3	5e-2
Hindsight residual baseline cost	1e-2	5e-3	—
Hindsight classifier cost	1e-2	1e-2	—
Action independence cost	1e1	1e1	—
Learning rate	5e-4	5e-4	1e-3

Table 4: Task Interleaving hyperparameters

C. Relation between HCA, CCA, and FC estimators

The FC estimators generalize both the HCA and CCA estimators. From FC, we can derive CCA by assuming that Φ_t and A_t are conditionally independent (see next section). We can also derive state and return HCA from FC.

For return HCA, we obtain both an all-action and baseline version of return HCA by choosing $\Phi_t = G_t$. For state HCA, we first need to decompose the return into a sum of rewards, and apply the policy gradient estimator to each reward separately. For a pair (X_t, R_{t+k}), and assuming that R_{t+k} is a function of X_{t+k} for simplicity, we choose $\Phi_t = X_{t+k}$. We then sum the different FC estimators for different values of k and obtain both an all-action and single-action version of state HCA.

Note however that HCA and CCA *cannot* be derived from one another. Both estimators leverage different approaches for unbiasedness, one (HCA) leveraging importance sampling, and the other (CCA) eschewing importance sampling in favor of constraint satisfaction (in the context of inference, this is similar to the difference between obtaining samples of the posterior by importance sampling versus directly parametrizing the posterior distribution).

D. Proofs

D.1. Policy gradients

Proof of equation 1. By linearity of expectation, the expected return can be written as $\mathbb{E}[G] = \sum_t \gamma^t \mathbb{E}[R_t]$. Writing the expectation as an integral over trajectories, we have:

$$\mathbb{E}[R_t] = \sum_{\substack{x_0,\ldots,x_t \\ a_0,\ldots,a_t}} \left(\prod_{s\leq t} \left(\pi_\theta(a_s|x_s) P(x_{s+1}|x_s,a_s)\right) \right) R(x_t,a_t)$$

Taking the gradient with respect to θ:

$$\nabla_\theta \mathbb{E}[R_t] \quad = \quad \sum_{\substack{x_0,\ldots,x_t \\ a_0,\ldots,a_t}} \left(\sum_{s'\leq t} \nabla_\theta \pi_\theta(a_{s'}|x_{s'}) P(x_{s'+1}|x_{s'},a_{s'}) \left(\prod_{s\leq t, s\neq s'} \left(\pi_\theta(a_s|x_s) P(x_{s+1}|x_s,a_s)\right) \right) \right) R(x_t,a_t)$$

We then rewrite $\nabla_\theta \pi_\theta(a_{s'}|x_{s'}) = \nabla_\theta \log \pi_\theta(a_{s'}|x_{s'}) \pi_\theta(a_{s'}|x_{s'})$, and obtain

$$\begin{aligned}
\nabla_\theta \mathbb{E}[R_t] &= \sum_{\substack{x_0,\ldots,x_t \\ a_0,\ldots,a_t}} \left(\sum_{s'\leq t} \nabla_\theta \pi_\theta(a_{s'}|x_{s'}) \left(\prod_{s\leq t, s} \left(\pi_\theta(a_s|x_s) P(x_{s+1}|x_s,a_s)\right) \right) \right) R(x_t,a_t) \\
&= \mathbb{E}\left[\sum_{s'\leq t} \nabla_\theta \log \pi_\theta(A_{s'}|X_{s'}) R_t \right]
\end{aligned}$$

Summing over t, we obtain

$$\nabla_\theta \mathbb{E}[G] = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t \sum_{s'\leq t} \nabla_\theta \log \pi_\theta(A_{s'}|X_{s'}) R_t \right]$$

which can be rewritten (with a change of variables):

$$\begin{aligned}
\nabla_\theta \mathbb{E}[G] &= \mathbb{E}\left[\sum_{t\geq 0} \nabla_\theta \log \pi_\theta(A_t|X_t) \sum_{t'\geq t} \gamma^{t'} R_{t'} \right] \\
&= \mathbb{E}\left[\sum_{t\geq 0} \gamma^t \nabla_\theta \log \pi_\theta(A_t|X_t) \sum_{t'\geq t} \gamma^{t'-t} R_{t'} \right] \\
&= \mathbb{E}\left[\sum_{t\geq 0} \gamma^t S_t G_t \right]
\end{aligned}$$

To complete the proof, we need to show that $\mathbb{E}[S_tV(X_t)] = 0$. By iterated expectation, $\mathbb{E}[S_tV(X_t)] = \mathbb{E}[\mathbb{E}[S_tV(X_t)|X_t]] = \mathbb{E}[V(X_t)\mathbb{E}[S_t|X_t]]$, and we have $\mathbb{E}[S_t|X_t] = \sum_a \nabla_\theta \pi(a|X_t) = \nabla_\theta(\sum_a \pi(a|X_t)) = \nabla_\theta 1 = 0$. □

Proof of equation 2. We start from the single action policy gradient $\nabla_\theta \mathbb{E}[G] = \mathbb{E}\left[\sum_{t\geq 0}\gamma^t S_t G_t\right]$ and analyse the term for time t, $\mathbb{E}[S_tG_t]$.

$$\begin{aligned}\mathbb{E}[S_tG_t] =&\mathbb{E}[\mathbb{E}[S_tG_t|X_t, A_t]]\\ =&\mathbb{E}[S_t\mathbb{E}[G_t|X_t, A_t]]\\ =&\mathbb{E}[S_tQ(X_t, A_t)]\\ =&\mathbb{E}\left[\mathbb{E}[S_tQ(X_t, A_t)|X_t]\right]\\ =&\mathbb{E}\left[\sum_a \nabla_\theta \pi_\theta(a|X_t)Q(X_t, a)\right]\end{aligned}$$

The first and fourth inequality come from different applications of iterated expectations, the second from the fact S_t is a constant conditional on X_t, A_t, and the third from the definition of $Q(X_t, A_t)$. □

D.2. Proof of FC-PG theorem

Proof of theorem 1 (single action). We need to show that $\mathbb{E}\left[S_t\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t,\Phi_t)}V(X_t, \Phi_t)\right] = 0$, so that $\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t,\Phi_t)}V(X_t, \Phi_t)$ is a valid baseline. As previously, we proceed with the law of iterated expectations, by conditioning successively on X_t then Φ_t

$$\begin{aligned}\mathbb{E}\left[S_t\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t, \Phi_t)}V(X_t, \Phi_t)\right] =&\mathbb{E}\left[\mathbb{E}\left[S_t\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t, \Phi_t)}V(X_t, \Phi_t)\middle|X_t, \Phi_t\right]\right]\\ =&\mathbb{E}\left[V(X_t, \Phi_t)\mathbb{E}\left[S_t\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t, \Phi_t)}\middle|X_t, \Phi_t\right]\right]\end{aligned}$$

Then we note that

$$\begin{aligned}\mathbb{E}\left[S_t\frac{\pi(A_t|X_t)}{\mathbb{P}(A_t|X_t, \Phi_t)}\middle|X_t, \Phi_t\right] &= \sum_a \mathbb{P}(a|X_t, \Phi_t)\nabla \log \pi(a|X_t)\frac{\pi(a|X_t)}{\mathbb{P}(a|X_t, \Phi_t)}\\ &= \sum_a \nabla\pi(a|X_t) = 0.\end{aligned}$$

□

Proof of theorem 1 (all-action). We start from the definition of the Q function:

$$\begin{aligned}Q(X_t, a) &= \mathbb{E}\left[G_t|X_t, A_t = a\right]\\ &= \mathbb{E}_{\Phi_t}\left[\mathbb{E}\left[G_t|X_t, \Phi_t, A_t = a\right]|X_t, A_t = a\right]\\ &= \int_\phi \mathbb{P}(\Phi = \varphi|X_t, A_t = a)Q(X_t, \Phi_t = \varphi, a)\end{aligned}$$

We also have

$$\mathbb{P}(\Phi = \varphi|X_t, A_t) = \frac{\mathbb{P}(\Phi = \varphi|X_t)\mathbb{P}(A_t = a|X_t, \Phi_t = \phi)}{\mathbb{P}(A_t = a|X_t)},$$

which combined with the above, results in:

$$\begin{aligned}Q(X_t, a) =& \int_\phi \mathbb{P}(\Phi = \varphi|X_t)\frac{\mathbb{P}(A_t = a|X_t, \Phi_t = \phi)}{\pi_\theta(a|X_t)}Q(X_t, \Phi_t, a)\\ =&\mathbb{E}\left[\frac{\mathbb{P}(A_t = a|X_t, \Phi_t = \phi)}{\pi_\theta(a|X_t)}Q(X_t, \Phi_t, a)\middle|X_t\right]\end{aligned}$$

For the compatibility with policy gradient, we start from:

$$\mathbb{E}[S_t G_t] = \mathbb{E}\left[\sum_a \nabla_\theta \pi_\theta(a|X_t) Q(X_t, a)\right]$$

We replace $Q(X_t, a)$ by the expression above and obtain

$$\begin{aligned}
\mathbb{E}[S_t G_t] =& \mathbb{E}\bigg[\sum_a \nabla_\theta \pi_\theta(a|X_t) \mathbb{E}\bigg[\frac{\mathbb{P}(A_t = a|X_t, \Phi_t = \phi)}{\pi_\theta(a|X_t)} Q(X_t, \Phi_t, a)\bigg| X_t\bigg]\bigg] \\
=& \mathbb{E}\bigg[\mathbb{E}\bigg[\sum_a \nabla_\theta \pi_\theta(a|X_t) \frac{\mathbb{P}(A_t = a|X_t, \Phi_t = \phi)}{\pi_\theta(a|X_t)} Q(X_t, \Phi_t, a)\bigg| X_t\bigg]\bigg] \\
=& \mathbb{E}\bigg[\mathbb{E}\bigg[\sum_a \nabla_\theta \log \pi_\theta(a|X_t) \mathbb{P}(A_t = a|X_t, \Phi_t = \phi) Q(X_t, \Phi_t, a)\bigg| X_t\bigg]\bigg] \\
=& \mathbb{E}\bigg[\sum_a \nabla_\theta \log \pi_\theta(a|X_t) \mathbb{P}(A_t = a|X_t, \Phi_t = \phi) Q(X_t, \Phi_t, a)\bigg]
\end{aligned}$$

Note that in the case of a large number of actions, the above can be estimated by

$$\frac{\nabla_\theta \log \pi_\theta(A'_t|X_t) \mathbb{P}(A'_t|X_t, \Phi_t = \phi)}{\pi_\theta(A'_t|X_t)} Q(X_t, \Phi_t, A'_t),$$

where A'_t is an independent sample from $\pi(.|X_t)$; note in particular that A'_t shall NOT be the action A_t that gave rise to Φ_t, which would result in a biased estimator.

D.3. Proof of CCA-PG theorem

Assume that Φ_t and A_t are conditionally independent on X_t. Then, $\frac{\mathbb{P}(A_t=a|X_t,\Phi_t=\phi)}{\mathbb{P}(A_t=a|X_t)} = 1$. In particular, it is true when evaluating at the random value A_t. From this simple observation, both CCA-PG theorems follow from the FC-PG theorems.

To prove the lower variance of the hindsight advantage estimate, note that

$$\begin{aligned}
\mathbb{V}[G_t - V(X_t, \Phi)] =& \mathbb{E}[(G_t - V(X_t, \Phi_t))^2] \\
=& \mathbb{E}[G_t^2] - \mathbb{E}[V(X_t, \Phi_t)^2] \\
\mathbb{V}[G_t - V(X_t)] =& \mathbb{E}[(G_t - V(X_t))^2] \\
=& \mathbb{E}[G_t^2] - \mathbb{E}[V(X_t)^2]
\end{aligned}$$

To prove the first statement, we have $(G_t - V(X_t, \Phi_t))^2 = G_t^2 + V(X_t, \Phi_t)^2 - 2G_t V(X_t, \Phi_t)$, and apply the law of iterated expectations to the last term:

$$\begin{aligned}
\mathbb{E}[G_t V(X_t, \Phi_t)] =& \mathbb{E}[\mathbb{E}[G_t V(X_t, \Phi_t)|X_t, \Phi_t]] \\
=& \mathbb{E}[V(X_t, \Phi_t)\mathbb{E}[G_t|X_t, \Phi_t]] \\
=& \mathbb{E}[V(X_t, \Phi_t)^2]
\end{aligned}$$

The proof for the second statement is identical. Finally, we note that by Jensen's inequality, we have $\mathbb{E}[V(X_t, \Phi_t)^2] \leq \mathbb{E}[V(X_t)^2]$, from which we conclude that $\mathbb{V}[G_t - V(X_t, \Phi_t)] \leq \mathbb{V}[G_t - V(X_t)]$.

□

E. Variance analysis

E.1. Relation between variance of advantage and variance of policy gradient

Consider an advantage estimate Y_t, i.e. a variable such that $\mathbb{E}[Y_t|X_t = x, A_t = a] = Q(x, a) - V(x)$. Possible choices for Y_t include the CCA estimate $G_t - V(X_t, \Phi_t)$ as well as the actual advantage $\mathcal{A}(x, a) = Q(x, a) - V(x)$. Note that

$\nabla_\theta \mathbb{E}[G_t] = \mathbb{E}[\sum_t \gamma^t S_t Y_t]$. We aim to analyze the variance of a single term $S_t Y_t$ (understanding the variance of the sum is more involved). More precisely, we compare the variance $\mathbb{V}[S_t Y_t | X_t]$ of the policy gradient term $S_t Y_t$ given X_t when using Y_t to that of $S_t \mathcal{A}_t$.

We use the conditional variance formula:

$$\mathbb{V}[S_t Y_t | X_t] = \mathbb{E}[\mathbb{V}[S_t Y_t | X_t, A_t] | X_t] + \mathbb{V}[\mathbb{E}[S_t Y_t | X_t, A_t] | X_t],$$

where S_t is constant given X_t, A_t. Therefore the first term becomes $\mathbb{E}[S_t^2 \mathbb{V}[Y_t | X_t, A_t] | X_t]$, and the second one $\mathbb{V}[S_t \mathbb{E}[Y_t | X_t, A_t] | X_t] = \mathbb{V}[S_t \mathcal{A}_t | X_t]$; this term does not depend on the actual advantage estimate used - it is equal to the variance of the policy gradient estimate when using the exact advantage $\mathcal{A}_t$. The additional variance incurred by using an unbiased advantage estimate Y_t instead of the exact advantage $\mathcal{A}_t$ is therefore:

$$\mathbb{V}[S_t Y_t | X_t] - \mathbb{V}[S_t \mathcal{A}_t | X_t] = \mathbb{E}[S_t^2 \mathbb{V}[Y_t | X_t, A_t] | X_t].$$

We see that the (conditional) advantage variance $\mathbb{V}[Y_t | X_t, A_t]$ (as well as the variance of the score function, and their correlation) drives the variance of the policy gradient estimator. We can further find a loose upper bound purely in terms of the unconditional variance of the advantage. First, suppose that the actions are discrete, and that the action distribution is parametrized by a softmax over logits $l_1, \ldots, l_k$, where k is the number of actions. Note that the score is $S_t = \frac{\partial \log \pi(A_t)}{\partial \theta} = \sum_{a'} \frac{\partial \log \pi(A_t)}{\partial l_{a'}} \frac{\partial l_{a'}}{\partial \theta}$, so

$$\begin{aligned} |S_t| =& |\sum_{a'} \frac{\partial \log \pi(A_t)}{\partial l_{a'}} \frac{\partial l_{a'}}{\partial \theta}| \\ \leq& \sum_{a'} |\frac{\partial \log \pi(A_t)}{\partial l_{a'}}| |\frac{\partial l_{a'}}{\partial \theta}| \\ \leq& \sum_{a'} |\frac{\partial l_{a'}}{\partial \theta}| \leq ||J||_1 \end{aligned}$$

where J is the jacobian of the function mapping parameters θ to logits. The second inequality is due to $|\frac{\partial \log \pi(a)}{\partial l_{a'}}| = |\delta_{a,a'} - \pi(a')| \leq 1$. It follows that:

$$\begin{aligned} \mathbb{V}[S_t Y_t | X_t] - \mathbb{V}[S_t \mathcal{A}_t | X_t] \leq& ||J||_1^2 \, \mathbb{E}[\mathbb{V}[Y_t | X_t, \mathcal{A}_t] | X_t] \\ \leq& ||J||_1^2 \, (\mathbb{V}[Y_t | X_t] - \mathbb{V}[\mathcal{A}_t | X_t])) \end{aligned}$$

again using the law of conditional variance $\mathbb{V}[Y_t | X_t] = \mathbb{V}[\mathbb{E}[Y_t | X_t, \mathcal{A}_t] | X_t] + \mathbb{E}[\mathbb{V}[Y_t | X_t, \mathcal{A}_t] | X_t]$. We thus see that the excess variance incurred by using Y_t in the policy gradient estimate can be upper bounded by a constant times the excess variance of the advantage estimate.

E.2. Variance analysis in the bandit problem

Here we provide a back-of-the-envelope variance analysis of the bandit problem. For simplicity (but reasoning can easily be extended), we assume no context and only two actions $\{0, 1\}$, and three vectors $W, V_0, V_1 \in \mathbb{R}^K$ (randomly sampled from a Gaussian and kept constant across all episodes). ϵ_r and ϵ_f are the reward and observation noise respectively, with standard deviations $\sigma_r \gg \sigma_f$. The feedback vector for action a is $W\epsilon_r + V_a + \epsilon_f$.

A forward (in this case, constant) baseline for this problem will have square advantage roughly scale as a σ_r^2.

Let's consider linear hindsight baseline $\alpha^T F$, which is equal to $\epsilon_r(\alpha^T W) + \alpha^T V_a + \alpha^T \epsilon_f$. The expected square advantage $\mathbb{E}[(G - \alpha^T F)^2]$ is therefore

$$\begin{aligned} \mathbb{E}[(G - \alpha^T F)^2] =& \pi_0 \mathbb{E}[\big(\epsilon_r(\alpha^T W - 1) + \alpha^T V_0 + \alpha^T \epsilon_f\big)^2] + \pi_1 \mathbb{E}[\big(\epsilon_r(\alpha^T W - 1) + (\alpha^T V_1 - 1) + \alpha^T \epsilon_f\big)^2] \\ =& (\alpha^T W - 1)^2 \sigma_r^2 + (\alpha^T \alpha) \sigma_f^2 + \pi_0 (\alpha^T V_0)^2 + \pi_1 (\alpha^T V_1 - 1)^2 \end{aligned}$$

The vectors W, V_0 and V_1 are independent with probability one (in fact they are nearly orthogonal), one can find a hindsight baseline such that $\alpha^T W - 1 = \alpha^T V_0 = \alpha^T V_1 - 1 = 0$, which leaves an expected squared advantage of $\sigma_f^2 \alpha^T \alpha$ which is small (for random vectors the matrices will be well-conditioned, the resulting α will have small norm); however that advantage leads to a biased update since the advantage is independent of the action. However, choosing $\alpha^T W = 1$ but $\alpha^T V_0 = \alpha^T V_1 = 0$ leads to a hindsight baseline which is equal to $\epsilon_r + \alpha^T \epsilon_f$, independent from the action; the effect of the noise ϵ_r will be removed entirely from the squared advantage, leading to an unbiased gradient estimator with a considerably lower variance (of order σ_f^2).

F. RL algorithms, common randomness, structural causal models

In this section, we provide an alternative view and intuition behind the CCA-PG algorithm by investigating credit assignment through the lens of causality theory, in particular *structural causal models* (SCMs) (Pearl, 2009a). These ideas are very related to the use of common random numbers (CRN), a standard technique in optimization with simulators (Glasserman & Yao, 1992).

F.1. Structural causal model of the MDP

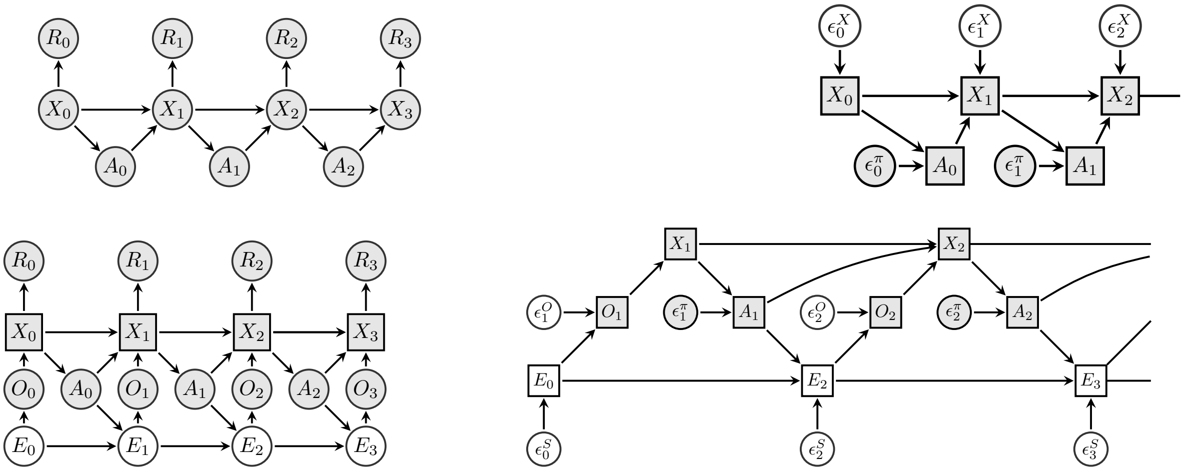

Figure 14: Graphical models and corresponding SCMs for RL problems. Top: MDP, bottom: POMDP; left: graphical model, right: structural causal model. Squares represent deterministic nodes, while circles represent stochastic nodes. Observed nodes are shaded in gray.

Structural causal models (SCM) (Pearl, 2009a) are, informally, models where all randomness is exogenous, and where all variables of interest are modeled as deterministic functions of other variables and of the exogenous randomness. They are of particular interest in causal inference as they enable reasoning about interventions, i.e. how would the *distribution* of a variable change under external influence (such as forcing a variable to take a given value, or changing the process that defines a variable), and about counterfactual interventions, i.e. how would a particular observed outcome (sample) of a variable have changed under external influence. Formally, a SCM is a collection of model variables $\{V \in \boldsymbol{V}\}$, exogenous random variables $\{\mathcal{E} \in \boldsymbol{\mathcal{E}}\}$, and distributions $\{p_{\mathcal{E}}(\varepsilon), \mathcal{E} \in \boldsymbol{\mathcal{E}}\}$, one per exogenous variable, and where the exogenous random variables are all assumed to be independent. Each variable V is defined by a function $V = f_V(\text{pa}(V), \boldsymbol{\mathcal{E}})$, where $\text{pa}(V)$ is a subset of $\boldsymbol{V}$ called the parents of V. The model can be represented by a directed graph in which every node has an incoming edge from each of its parents. For the SCM to be valid, the induced graph has to be a directed acyclic graph (DAG), i.e. there exists a topological ordering of the variables such that for any variable V_i, $\text{pa}(V_i) \subset \{V_1, \ldots, V_{i-1}\}$; in the following we will assume such an ordering. This provides a simple sampling mechanism for the model, where the exogenous random variables are first sampled according to their distribution, and each node is then computed in topological order. Note that any probabilistic model can be represented as a SCM by virtue of reparametrization (Kingma & Ba, 2014; Buesing et al., 2019). However, such a representation is not unique, i.e. different SCMs can induce the same distribution.

In the following we give an SCM representation of a MDP (see Fig.14 for the causal graphical model and corresponding SCM for MDPs and POMDPs). The transition from X_t to X_{t+1} under A_t is given by the transition function f^X: $X_{t+1} = f^X(X_t, A_t, \mathcal{E}_t^X)$ with exogenous variable / random number $\mathcal{E}_t^X$. The policy function f^π maps a random number $\mathcal{E}_t^\pi$, policy parameters θ, and current state X_t to the action $A_t = f^\pi(X_t, \mathcal{E}_t^\pi, \theta)$. Together, f^π and $\mathcal{E}_t^\pi$ induce the policy, a distribution $\pi_\theta(A_t|X_t)$ over actions. Without loss of generality we assume that the reward is a deterministic function of the state and action: $R_t = f^R(X_t, A_t)$. $\mathcal{E}^X$ and $\mathcal{E}^\pi$ are random variables with a fixed distribution; all changes to the policy are absorbed by changes to the deterministic function f^π. Denoting $\mathcal{E}_t = (\mathcal{E}^X_t, \mathcal{E}^\pi_t)$, note the next reward and state (X_{t+1}, R_t) are deterministic functions of X_t and $\mathcal{E}_t$, since we have $X_{t+1} = f^X(X_t, f^\pi(X_t, \mathcal{E}^\pi_t, \theta), \mathcal{E}^X_t)$ and similarly $R_t = R(X_t, f^\pi(X_t, \mathcal{E}^\pi_t, \theta)$. Let $X_{t+} = (X_{t'})_{t'>t}$ and similarly, $\mathcal{E}_{t+} = (\mathcal{E}^X_t, \mathcal{E}_{t'})_{t'>t}$ Through the composition of the functions f^X, f^π and R, the return G_t (under policy π) is a deterministic function f^G of X_t, A_t and $\mathcal{E}_{t+}$.

F.2. Proof of theorem 2

For notation purposes, in the rest of this section, we will focus on credit assignment for action A_t (since policy gradient terms are additive with respect to time), and will denote $X = X_t$, $A = A_t$, $\varepsilon = \mathcal{E}_{t^+}$, and $\tau = (X_s, R_r, A_s)_{s\geq t}$. Furthermore, we will denote $\Phi = \Phi_t$.

From the arguments in the section above, one can write $\tau = f^\tau(X, A, \varepsilon)$, $G = f^G(\tau)$, and $\Phi = f^\phi(\tau)$. We may integrate out τ, in which case the graph only contains X, A, ε and G. In that graph, by the faithfulness assumption, there can be no causal path from A to Φ, as this would violate the conditional independence assumption. It follows that there are functions g_G and g_Φ such that $G = g_G(X, A, \varepsilon)$ and $\Phi = g_\Phi(X, \varepsilon)$.

The resulting structural causal models can be seen in Fig. 15.

The conditional expectation $Q(x, a, \phi)$ is given by $Q(x, a, \phi) = \int_\varepsilon p(\varepsilon|x, \phi, a)G(x, a, \varepsilon)$ The counterfactual return for action a, having observed ϕ is given by $\mathbb{E}[G(\tau')|\tau' \sim P(\tau'|X = x, \text{observe}(\Phi = \phi))]$ is equal to $\int_\varepsilon p(\varepsilon|x, \phi)G(x, a, \phi)$.

Finally, note that from d-separation $p(\varepsilon|x, \phi) = p(\varepsilon|x, a, \phi)$, and the result follows.

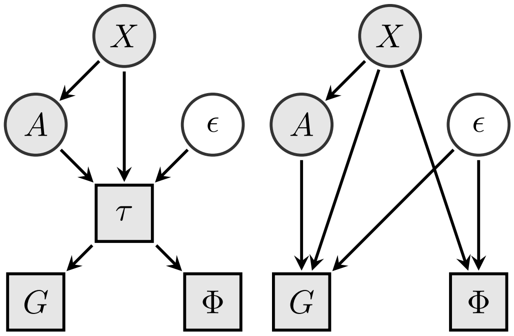

Figure 15: SCMs for the reduced action selection problem; left: including the trajectory; right: trajectory is integrated out. There is no arrow from A to Φ on the right since the graph is assumed to be faithful and A and Φ are conditionally independent given X.

G. Individual Treatment Effects, (Conditional) Average Treatment Effects, Counterfactuals and Counterfactual identifiability

In this section, we will further link the ideas developed in this report to causality theory. In particular we will connect them to two notions of causality theory known as individual treatment effect (ITE) and average treatment effect (ATE). In the previous section, we extensively leveraged the framework of structural causal models. It is however known that distinct SCMs may correspond to the same distribution; learning a model from data, we may learn a model with correct distribution but with with incorrect structural parametrization and counterfactuals. We may therefore wonder whether counterfactual-based approaches may be flawed when using such a model. We investigate this question, and analyze our algorithm in very simple settings for which closed-form computations can be worked out.

G.1. Individual and Average Treatment Effects

Consider a simple medical example which we model with an SCM as illustrated in Fig. 16. We assume population of patients, each with a full medical state denoted S, which summarizes all factors, known or unknown, which affect a patient's future health such as genotype, phenotype etc. While S is never known perfectly, some of the patient's medical history H may be known, including current symptoms. On the basis of H, a treatment decision T is taken; as is often done, for simplicity we consider T to be a binary variable taking values in {1='treatment', 0='no treatment'}. Finally, health state S and treatment T result in a observed medical outcome O, a binary variable taking values in {1='cured', 0='not cured'}. For a given value $S = s$ and $T = t$, the outcome is a function (also denoted O for simplicity) $O(s, t)$. Additional medical information F may be observed, e.g. further symptoms or information obtained after the treatment, from tests such as X-rays, blood tests, or autopsy.

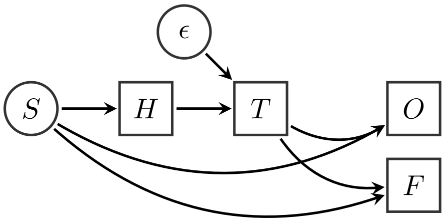

Figure 16: The medical treatment example as a structured causal model.

In this simple setting, we can charactertize the effectiveness of the treatment for an individual a patient with profile S by the Individual Treatment Effect (ITE) which is defined as the difference between the outcome under treatment and no treatment.

Definition 1 (Individual Treatment Effect)**.**

$$\begin{aligned} \textit{ITE(s)} =& \mathbb{E}[O|S = s, \mathrm{do}(T = 1)] - \mathbb{E}[O|S = s, \mathrm{do}(T = 0)] \\ =& O(s, T = 1) - O(s, T = 0) \end{aligned} \tag{8}$$

The conditional average treatment effect is the difference in outcome between the choice of $T = 1$ and $T = 0$ when averaging over all patients with the same set of symptoms $H = h$

Definition 2 (Conditional Average Treatment Effect)**.**

$$\begin{aligned} \textit{ATE}(h) =& \mathbb{E}[O|H = h, \mathrm{do}(T = 1)] - \mathbb{E}[O|H = h, \mathrm{do}(T = 0)] \\ =& \int_s p(S = s|H = h)(O(s, T = 1) - O(s, T = 0)) \end{aligned} \tag{9}$$

Since the exogenous noise (here, S) is generally not known, the ITE is typically an unknowable quantity. For a particular patient (with hidden state S), we will only observe the outcome under $T = 0$ or $T = 1$, depending on which treatment option was chosen; the counterfactual outcome will typically be unknown. Nevertheless, for a given SCM, it can be counterfactually estimated from the outcome and feedback.

Definition 3 (Counterfactually Estimated Individual Treatment Effect)**.**

$$\textit{CF-ITE}[H = h, F = f, T = 1] = \delta(o = 1) - \int_{s'} P(S = s'|H = h, F = f, T = 1)O(s', T = 0) \tag{10}$$

$$\textit{CF-ITE}[H = h, F = f, T = 0] = \int_{s'} P(S = s'|H = h, F = f, T = 1)O(s', T = 0) - \delta(o = 1) \tag{11}$$

In general the counterfactually estimated ITE will not be exactly the ITE, since there may be remaining uncertainty on s. However, the following statements relate CF-ITE, ITE and ATE:

- If S is identifiable from O and F with probability one, then the counterfactually-estimated ITE is equal to the ITE.
- The average (over S, conditional on H) of the ITE is equal to the ATE.
- The average (over S and F, conditional on H) of CF-ITE is equal to the ATE.

Assimilating O to a reward, the above illustrates that the ATE (equation 9) essentially corresponds to a difference of Q functions, the ITE (equation 8) to a difference of returns under common randomness, and the counterfactual ITE to CCA-like advantage estimates. In contrast, the advantage estimate $G_t - V(H_t)$ is a difference between a return (a sample-level quantity) and a value function (a population-level quantity, which averages over all individuals with the same medical history H); this discrepancy explains why the return-based advantage estimate can have very high variance.

As mentioned previously, for a given joint distribution over observations, rewards and actions, there may exist distinct SCMs that capture that distribution. Those SCMs will all have the same ATE, which measures the effectiveness of a policy on average. But they will generally have different ITE and counterfactual ITE, which, when using model-based counterfactual policy gradient estimators, will lead to different estimators. Choosing the 'wrong' SCM will lead to the wrong counterfactual, and so we may wonder if this is a cause for concern for our methods.

We argue that in terms of learning optimal behaviors (in expectation), estimating inaccurate counterfactual is not a cause for concern. Since all estimators have the same expectation, they would all lead to the correct estimates for the effect of switching a policy for another, and therefore, will all lead to the optimal policy given the information available to the agent. In fact, one could go further and argue that for the purpose of finding good policies in expectations, we should only care about the counterfactual for a precise patient inasmuch as it enables us to quickly and correctly taking better actions for future patients for whom the information available to make the decision (H) is very similar. This would encourage us to choose the SCM for which the CF-ITE has minimal variance, regardless of the value of the true counterfactual. In the next section, we elaborate on an example to highlight the difference in variance between different SCMs with the same distribution and optimal policy.

G.2. Betting against a fair coin

We begin from a simple example, borrowed from (Pearl, 2009b), to show that two SCMs that induce the same interventional and observational distributions can imply different counterfactual distributions. The example consists of a game to guess the outcome of a fair coin toss. The action A and state S both take their values in $\{h, t\}$. Under model **I**, the outcome O is 1 if $A = S$ and 0 otherwise. Under model **II**, the guess is ignored, and the outcome is simply $O = 1$ if $S = h$. For both models, the average treatment effect $E[O|A = h] - E[O|A = t]$ is 0 implying that in both models, one cannot do better than random guessing. Under model **I**, the counterfactual for having observed outcome $O = 1$ and changing the action, is always $O = 0$, and vice-versa (intuitively, changing the guess changes the outcome). Therefore, the ITE is ± 1. Under model **II**, all counterfactual outcomes are equal to the observed outcomes, since the action has in fact no effect on the outcome. The ITE is always 0.

In the next section, we will next adapt the medical example into a problem in which the choice of action does affect the outcome. Using the CF-ITE as an estimator for the ATE, we will find how the choice of the SCM affects the variance of that estimator (and therefore how the choice of the SCM should affect the speed at which we can learn which is the optimal treatment decision).

G.3. Medical example

Take the simplified medical example from Fig.16, where a population of patients with the same symptoms come to the doctor, and the doctor has a potential treatment T to administer. The state S represents the genetic profile of the patient, which can be one of three $\{\text{GENE}_\text{A}, \text{GENE}_\text{B}, \text{GENE}_\text{C}\}$ (each with probability $1/3$). We assume that genetic testing is not available and that we do not know the value of S for each patient. The doctor has to make a decision whether to administer drugs to this population or not, based on repeated experiments; in other words, they have to find out whether the average treatment effect is positive or not. We consider the two following models:

- In model **I**, patients of type GENE_A always recover, patients of type GENE_C never do, and patients of type GENE_B recover if they get the treatment, and not otherwise; in particular, in this model, administering the drug never hurts.
- In model **II**, patients of type GENE_A and GENE_B recover when given the drug, but not patients of type GENE_C; the situation is reversed (GENE_A and GENE_B patients do not recover, GENE_C do) when not taking the drug.

In both models - the true value of giving the drug is $2/3$, and not giving the drug $1/3$, which leads to an ATE of $1/3$. For each model, we will evaluate the variance of the CF-ITE, under one of the four possible treatment-outcome pair. The results are summarized in table 5. Under model **A**, the variance of the CF-ITE estimate (which is the variance of the advantage estimate used in CCA-PG gradient) is $1/6$, while it is 1 under model **B**, which would imply **A** is a better model to leverage counterfactuals into policy decisions.

Treatment	Outcome	Type	CF-Prob.		CF-O		ITE		CF-V		CF-ITE		Var
Drug	Cured	GENE$_\text{A}$	1/2	1/2	1	0	0	+1	1/2	0	1/2	1	1/6 1
		GENE$_\text{B}$	1/2	1/2	0	0	+1	+1					
		GENE$_\text{C}$	0	0	╳		╳						
	Not cured	GENE$_\text{A}$	0	0	╳		╳		0	1	0	-1	
		GENE$_\text{B}$	0	0	╳		╳						
		GENE$_\text{C}$	1	1	0	1	0	-1					
No Drug	Cured	GENE$_\text{A}$	1	0	1	1	0	0	1	0	0	1	1/6 1
		GENE$_\text{B}$	0	0	╳		╳						
		GENE$_\text{C}$	0	1	0	0	1	1					
	Not cured	GENE$_\text{A}$	0	1/2	1	1	-1	-1	1/2	1	-1/2	-1	
		GENE$_\text{B}$	1/2	1/2	1	1	-1	-1					
		GENE$_\text{C}$	1/2	0	0	0	0	0					

Table 5: CCA-PG variance estimates in the medical example. CF-Probs. Red value are estimates for model **I**, blue ones are for model **II**. CF-Prob denotes posterior probabilities of the genetic state S given the treatment T and outcome O. CF-O is the counterfactual outcome. The ITE is the individual treatment effect (difference between outcome and counterfactual outcome). CF-V is the counterfactual value function, computed as the average of CF-O under the posterior probabilities for S. CF-ITE is the counterfactual advantage estimate (difference between O and CF-V). Var is the variance of CF-ITE under the prior probabilities for the outcome.