Only Relevant Information Matters: Filtering Out Noisy Samples to Boost RL

Yannis Flet-Berliac* and **Philippe Preux**

SequeL Inria
CRIStAL Univ. Lille
CNRS

Abstract

In reinforcement learning, policy gradient algorithms optimize the policy directly and rely on sampling efficiently an environment. Nevertheless, while most sampling procedures are based on direct policy sampling, self-performance measures could be used to improve such sampling prior to each policy update. Following this line of thought, we introduce SAUNA[1], a method where non-informative transitions are rejected from the gradient update. The level of information is estimated according to the fraction of variance explained by the value function $\mathcal{V}^{ex}$: a measure of the discrepancy between V and the empirical returns. In this work, we use this criterion to select samples that are useful to learn from, and we demonstrate that this selection can significantly improve the performance of policy gradient methods. In this paper: (a) We define $\mathcal{V}^{ex}$ and introduce the SAUNA method to filter transitions. (b) We conduct experiments on a set of benchmark continuous control problems. SAUNA significantly improves performance. (c) We investigate how $\mathcal{V}^{ex}$ reliably selects samples with the most positive impact on learning and study its improvement on both performance and sample efficiency.

1 Introduction

Learning to control agents in simulated environments has been a challenge for decades in reinforcement learning (RL) [Werbos, 1989; Robinson and Fallside, 1989; Schmidhuber and Huber, 1991] and has lately led to a lot of research efforts in this direction [Silver *et al.*, 2016; Ha and Schmidhuber, 2018; Espeholt *et al.*, 2018], notably in policy gradient methods [Silver *et al.*, 2014; Schulman *et al.*, 2016; Haarnoja *et al.*, 2018]. Despite progress, policy gradient algorithms still heavily suffer from sample inefficiency [Kakade, 2003; Wang *et al.*, 2017; Wu *et al.*, 2017]. In particular, many of those methods are subject to use as much experience as possible in the most efficient way. However, quantity is not quality: the quality of the sampling procedure also determines the learning curve of the agent and its final performance. Hence, we think that *not all experiences are worth using* in the gradient update. Indeed, some transitions may add noise to the gradient update, diluting relevant signals, and hindering learning. The central idea of SAUNA is to reject transitions that are not informative.

Use of non-informative or misinformative transitions can only mislead the learning process and waste computational time. Amari's natural gradient [Amari, 1998] concept concerns the geometry of the search space related to the "value of information": this has been studied for long in RL since [Kakade, 2002]. Our work focuses on a different notion of value of information, and treats it differently: we evaluate whether a transition conveys useful information and use it only if it is considered beneficial to learning. For this purpose, we use a measure of the discrepancy between the estimated state value and the observed returns. This discrepancy is formalized with the notion of the fraction of variance explained $\mathcal{V}^{ex}$ [Kvålseth, 1985]. Transitions for which $\mathcal{V}^{ex}$ is close to zero are those for which the correlation between the value function V and the observed returns is also close to zero. SAUNA keeps transitions where there is either a strong correlation or a lack of fit between V and the returns while avoiding the dilution of useful information by removing useless samples. We consider on-policy methods for their unbiasedness and stability compared to off-policy algorithms [Nachum *et al.*, 2017]. However, our method can be applied to off-policy methods as well, and we leave this investigation open for future work.

In summary, in this paper:

1. We propose to move from a traditional policy-based sampling procedure to a refined sample selection driven by $\mathcal{V}^{ex}$. We explore how transition filtering simplifies the underlying state space and affects performance.
2. We hypothesize that not all samples are useful for learning and that disturbing samples should be rejected to avoid performance loss. We provide experimental evidence corroborating this claim.
3. By combining (1) and (2), we obtain a learning algorithm that is empirically effective in learning neural net-

*Correspondence to: yannis.flet-berliac@inria.fr

[1] ***S**amples **A**re **U**seful? **N**ot **A**lways*. Saunas help to release impurities [noisy samples] and improve cell regeneration [PG update]. Their temperatures could be fatal if not regulated by humidity [$\mathcal{V}^{ex}$].

work policies for challenging control tasks. Our results significantly improve the state of the art in using RL for high-dimensional continuous control.

Section 2 recalls basic notions of policy gradient methods in RL and the notion of "fraction of variance explained" drawn from the statistics literature. Section 3 sets our contribution within the RL domain. Section 4 introduces SAUNA. Section 5 provides experimental evidence of the benefit of using our method, and also investigates various experimental aspects of SAUNA. Section 6 further discusses the method. Finally, Section 7 concludes and draws some lines of future research.

2 Preliminaries

2.1 Notations

We consider a Markov Decision Process (MDP) with states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, transition distribution $s_{t+1} \sim \mathcal{P}(s_t, a_t)$ and reward function $r_t \sim \mathcal{R}(s_t, a_t)$. Let $\pi(a|s)$ denote a stochastic policy and let the objective function be the expected sum of discounted rewards:

$$J(\pi) \triangleq \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r\left(s_t, a_t\right)\right], \tag{1}$$

where $\gamma \in [0, 1)$ is a discount factor [Puterman, 1994] and $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$ is a trajectory sampled from the environment while the agent is following a given policy π. Let us remind the notions of the value of a state in the MDP framework. The value $V^\pi(s)$ of a state s while following a policy π starting in state s is defined by: $V^\pi(s) \triangleq \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r\left(s_t, a_t\right)|s_0 = s\right]$.

Closely related is the value (or quality) of a state-action pair: the quality $Q^\pi(s, a)$ of performing action a in state s and then following policy π is defined by: $Q^\pi(s, a) \triangleq \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r\left(s_t, a_t\right)|s_0 = s, a_0 = a\right]$. Finally, the advantage function quantifies how an action a is better than the average action in state s (following policy π): $A^\pi(s, a) \triangleq Q^\pi(s, a) - V^\pi(s)$. MDP theory asserts that there exists an optimal policy π^* that maximizes J: we denote its value function V^*. In practice, value functions are unknown; we denote V, Q, and A their current estimates.

2.2 Policy Gradient Methods

Policy gradient methods aim at optimizing the policy directly [Williams, 1992]. The policy π is often implemented with a function parameterized by θ: learning a policy boils down to finding the best parameters. In the sequel, we use θ to denote the parameters as well as the policy. In deep RL, the policy is represented by a neural network (the policy network) and is assumed to be continuously differentiable with respect to its parameters θ. When the policy is represented by such a parameterized function, hence by an approximation of a policy, the MDP theory basically breaks down.

In this paper, we consider Proximal Policy Optimization (PPO) [Schulman *et al.*, 2017], an on-policy policy gradient method achieving state of the art performance on a suite of benchmark tasks despite a relatively simple implementation. Very interestingly, PPO is an evolution of TRPO that builds on the notion of natural gradient, hence Amari's notion of "value of information" mentioned above. PPO has been shown to outperform TRPO experimentally. By building on PPO, this paper combines two different ideas related to the notion of the value of information. At each episode, PPO collects (s_t, a_t, r_t) samples using its current policy θ_k. After some episodes, using these collected transitions, PPO updates its policy and gets a new one θ_{k+1}:

$$\theta_{k+1} \leftarrow \mathop{\text{argmax}}_{\theta} \mathop{\mathbb{E}}_{s_t, a_t\sim\pi_{\theta_k}}\left[\mathcal{L}_{\text{PPO}}\left(s_t, a_t, \theta_k, \theta\right)\right]. \tag{2}$$

We use the clipped version of PPO:

$$\mathcal{L}_{\text{PPO}}(s_t, a_t, \theta_k, \theta) = \text{Clip}(A^{\pi_{\theta_k}}(s_t, a_t), \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, \delta), \tag{3}$$

where $\text{Clip}(A, \alpha, \delta) = \begin{cases} \min\left(\alpha A, (1+\delta)A\right), A \geq 0 \\ \min\left(\alpha A, (1-\delta)A\right), A < 0. \end{cases}$ A is the advantage function introduced above. Clipping makes the training updates more stable: it ensures that the gradient steps do not lead the policy outside of the region of parameter space where the samples collected are informative.

2.3 $\mathcal{V}^{ex}$: Fraction of Variance Explained

Now we introduce the key notion of this paper, namely the fraction of variance explained, denoted by $\mathcal{V}^{ex}$. As shown in this paper, a yet elementary use of this concept strikingly improves the performance of policy gradient algorithms. In general terms, $\mathcal{V}^{ex}$ gives some information about the goodness of fit of a model. In statistics, it is also denoted R^2, which is a poor notation since this quantity can be negative for non-linear models [Kvålseth, 1985] (also, in the context of RL, R usually refers to the return). This quantity is also known as the coefficient of determination. In a regression setting, assume a model $\hat{y}$ aims at predicting y from x, given a set of N couples (x_i, y_i), $\mathcal{V}^{ex}$ is defined by:

$$\mathcal{V}^{ex} \triangleq 1 - \frac{\text{MSE}}{\text{VAR}} \tag{4}$$

where MSE is the mean squared error of the model measured on these N couples ($\text{MSE} = \frac{1}{N}\sum_i \left(y_i - \hat{y}(x_i)\right)^2$), and VAR is the variance of the observed targets y_i. $\mathcal{V}^{ex} \leq 1$ and:

- $\mathcal{V}^{ex} = 1$ means that the model perfectly predicts the data (MSE $= 0$).
- $\mathcal{V}^{ex} = 0$ means that the model performs as always predicting the average (MSE $=$ VAR).
- $\mathcal{V}^{ex} < 0$ means that the model performs worse than merely predicting the mean value (MSE $>$ VAR).

3 Related Work

Our method integrates three key ideas: (a) function approximation with a neural network combining or separating the actor and the critic with an on-policy setting, (b) transition filtering reducing information/signal dilution in the gradient update while simplifying the underlying MDP, and (c) using $\mathcal{V}^{ex}$ as a measure of correlation between the value function

and the returns to allow better sampling and more efficient learning. Below, we consider previous work building on some of these approaches.

Actor-critic algorithms essentially use the value function to alternate between policy evaluation and policy improvement [Andrew *et al.*, 1983; Sutton and Barto, 2018]. In order to update the actor, many methods adopt the on-policy formulation [Peters and Schaal, 2008; Mnih *et al.*, 2016; Schulman *et al.*, 2017]. However, despite their important successes, these methods suffer from sample complexity.

In the literature, research has also been conducted in prioritization sampling. While [Schaul *et al.*, 2016] makes the learning from experience replay more efficient by using the TD error as a measure of these priorities in an off-policy setting, our method directly selects the samples on-policy. [Schmidhuber, 1991] is related to our method in that it calculates the expected improvement in prediction error, but with the objective to maximize the intrinsic reward through artificial curiosity. Instead, our method estimates the expected fraction of variance explained and filters out some of the samples to improve the learning efficiency.

$\mathcal{V}^{ex}$ has already been used in [Flet-Berliac and Preux, 2019] as one of the auxiliary tasks for self-assessment of performance. Finally, motion control in physics-based environments is a long-standing and active research field. In particular, there are many prior work on continuous action spaces [Levine and Abbeel, 2014; Heess *et al.*, 2015; Lillicrap *et al.*, 2016; Schulman *et al.*, 2016] that demonstrate how locomotion behavior and other skilled movements can emerge as the outcome of optimization problems.

4 SAUNA: Dynamic Transition Filtering

We introduce a general method to filter transitions that contain useful information for policy gradient updates. In this paper, we detail how to couple SAUNA with PPO, an on-policy gradient algorithm achieving state of the art performance. We refer to this combination as PPO+SAUNA. SAUNA can be coupled with other algorithms, especially with non-policy methods such as DQN: we leave this for future work. Below we detail how to adapt the notion of $\mathcal{V}^{ex}$ to RL.

4.1 $\mathcal{V}^{ex}$ applied to RL

The fraction of variance that the current estimate of the value function explains about the observed returns corresponds to the proportion of the variance in the dependent variable V that is predictable from s_t. We define $\mathcal{V}^{ex}_{\tau}$ as the fraction of variance explained for a trajectory τ:

$$\mathcal{V}^{ex}_{\tau} \triangleq 1 - \frac{\sum_{t\in\tau}\left(R_t - V(s_t)\right)^2}{\sum_{t\in\tau}\left(R_t - \langle R\rangle_\tau\right)^2}, \tag{5}$$

where $R_t = \sum_{k\geq 0}\gamma^k r_{t+k}$, r_t is the immediate reward collected at timestep t, $V(s_t)$ is the current estimate of the value of state s_t, and $\langle R\rangle_\tau$ is the average of the R_t in trajectory τ. This definition can be extended from a trajectory τ to a batch $\mathcal{B}$ of sampled transitions $\mathcal{V}^{ex}_{\mathcal{B}}$. In the RL context, the interpretation of $\mathcal{V}^{ex}_{\mathcal{B}}$ is:

- $\mathcal{V}^{ex}_{\mathcal{B}} = 1$: V perfectly explains the observed returns.
- $\mathcal{V}^{ex}_{\mathcal{B}} = 0$: V corresponds to a simple average prediction.
- $\mathcal{V}^{ex}_{\mathcal{B}} < 0$: V provides a worse prediction than the average of the returns.

The intuition is that $\mathcal{V}^{ex}_{\mathcal{B}}$ close to 1 corresponds to well-predicted returns. $\mathcal{V}^{ex}_{\mathcal{B}} < 0$ corresponds to a rather large prediction error of the value function, meaning that these samples are useful because the agent has something to learn from. On the other hand, $\mathcal{V}^{ex}_{\mathcal{B}}$ close to 0 means that the samples do not provide any valuable information to improve the value estimates. We will demonstrate that $\mathcal{V}^{ex}$ is indeed a relevant indicator for assessing self-performance in RL.

4.2 Estimating $\mathcal{V}^{ex}$

While sampling the environment, SAUNA rejects transitions for which $V(s_t)$ is not correlated with returns that have followed s_t. Therefore, $\mathcal{V}^{ex}_{\mathcal{B}}$ should be estimated at each timestep and we define $\mathcal{V}^{ex}_{\theta}(s_t)$ as the prediction of $\mathcal{V}^{ex}_{\mathcal{B}}$ with parameters θ at state $s_t \in \mathcal{B}$. In addition, for shared parameters configurations, an error term on the value estimation is added to the objective. The final objective function becomes:

$$\mathcal{L}_{\text{SAUNA}}(s_t, a_t, \theta_{old}, \theta) = \mathcal{L}_{\text{PPO}}(s_t, a_t, \theta_{old}, \theta) - \tag{6}$$

$$c_1\left(V_\theta(s_t) - R_t\right)^2 - \tag{7}$$

$$c_2\left(\mathcal{V}^{ex}_{\theta}(s_t) - \mathcal{V}^{ex}_{\mathcal{B}}\right)^2, \tag{8}$$

where c_1 and c_2 are the coefficients for the squared-error losses of respectively the value function and the fraction of variance explained function. Note that only the term (8) is specific to SAUNA. (6) and (7) come from PPO. When the network is not shared between the policy and the value function, SAUNA embeds $\mathcal{V}^{ex}_{\mathcal{B}}$ to the value function network using a single hidden layer. The rest of the network is unchanged, making our method very easy to use without significantly increasing the complexity of the underlying algorithm.

4.3 SAUNA Algorithm

Algorithm 1 shows the pseudocode of SAUNA when coupled with PPO. Overall, the resulting algorithm visits a set of trajectories along which it collects useful samples in the sense explained above, assessed with regards to $\mathcal{V}^{ex}$. The mechanism may be viewed as analogous to the method of dropout in deep learning [Srivastava *et al.*, 2014; Freeman *et al.*, 2019] although here dropout happens in the state space of the underlying MDP and is directed by $\mathcal{V}^{ex}$. Once a batch $\mathcal{B}$ of T such useful samples is collected, SAUNA performs the usual gradient update following the PPO template.

The gradient update concerns the three quantities estimated by SAUNA: the policy parameters θ, line 12, the value estimation parameters ϕ, line 13, and the $\mathcal{V}^{ex}$ estimation parameters ψ, line 14. The *if* statement filters the useful samples: $\widetilde{\mathcal{V}^{ex}_{\psi_k}}(s_{0:t-1})$ denotes the median of $\mathcal{V}^{ex}_{\psi_k}$ between timesteps 0 and $t-1$, ϵ_0 is a Laplace estimator (set to 10^{-8}), and ρ is the filtering threshold. One may legitimately ask why not use directly $|\mathcal{V}^{ex}_{\psi_k}(s_t)|$ in the predicate. The rationale is practical: the ratio is a standardized measure as the agent learns, stabilized by the median, more robust to outliers than the mean. For better legibility, Algorithm 1 does not share parameters between the π, V and $\mathcal{V}^{ex}$ networks. A version where these parameters would be partially shared is straightforward.

Algorithm 1 SAUNA coupled with PPO.

1: **Initialize** policy parameters θ_0, value function parameters ϕ_0 and $\mathcal{V}^{ex}$ function parameters ψ_0
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: $s_0 \leftarrow$ initial state
4: batch $\mathcal{B} \leftarrow \emptyset$
5: **while** size$(\mathcal{B}) \leq T$ **do**
6: $a_t \sim \pi_{\theta_k}(s_t)$
7: **execute** action a_t and observe r_{t+1} and s_{t+1}
8: **if** $\frac{|\mathcal{V}^{ex}_{\psi_k}(s_t)|}{|\mathcal{V}^{ex}_{\psi_k}(s_{0:t-1})|+\epsilon_0} \geq \rho$ **then**
9: **add** $\left(s_t, a_t, r_t, V_{\phi_k}(s_t), s_{t+1}, \mathcal{V}^{ex}_{\psi_k}(s_t)\right)$ to $\mathcal{B}$
10: **if** s_{t+1} is a final state **then**
11: $s_{t+1} \leftarrow$ initial state
12: $\theta_{k+1} \leftarrow \underset{\theta}{\text{argmax}} \sum_{t \in \mathcal{B}} \mathcal{L}_{\text{PPO}}(s_t, a_t, \theta_k, \theta)$
13: $\phi_{k+1} \leftarrow \underset{\phi}{\text{argmin}} \sum_{t \in \mathcal{B}} \left(V_\phi(s_t) - R_t\right)^2$
14: $\psi_{k+1} \leftarrow \underset{\psi}{\text{argmin}} \sum_{t \in \mathcal{B}} \left(\mathcal{V}^{ex}_{\psi}(s_t) - \mathcal{V}^{ex}_{\mathcal{B}}\right)^2$

5 Experiments

We have forked the *stable-baselines* repository [Hill *et al.*, 2018] and minimally modified the code to incorporate our method. Unless otherwise stated, the policy network used for all tasks is a fully-connected multi-layer perceptron with 2 hidden layers of 64 units. Moreover, the architecture for the $\mathcal{V}^{ex}$ function head is the same as for the value function head.

5.1 SAUNA in the Continuous Domain

To assess SAUNA, we compare PPO+SAUNA against its natural baseline PPO. We use six simulated robotic deterministic tasks from OpenAI Gym [Brockman *et al.*, 2016] using *MuJoCo* [Todorov *et al.*, 2012]. The two hyperparameters required by our method ($\rho = 0.3$ from Eq. 5 and $c_2 = 0.5$ from Eq. 8) and all the others (identical to those in [Schulman *et al.*, 2017]) are exactly the same for all tasks.

We made the choice of not tuning the hyperparameters for each algorithm and for each task to have a tougher assessment of SAUNA: only SAUNA-specific hyperparameters ρ and c_2 have been tuned by grid-search. Hence, the performance we report for SAUNA is not necessarily the best that could be obtained with parameter tuning. The graphs reported in Fig. 1 show that our method outperforms PPO on all considered continuous control tasks.

We then experiment with the more difficult, high-dimensional continuous domain environment of *Roboschool* [Klimov and Schulman, 2017] with various neural network sizes. In Fig. 2a, the same fully-connected network as for the previous *MuJoCo* experiments (2 hidden layers each with 64 neurons) is used. In Fig. 2b, the network is composed of a deeper and wider 3 hidden layers with 512, 256 and 128 neurons. We trained those agents with 32 parallel actors. In both experiments, PPO+SAUNA performs better and learns faster at the beginning. The gap closes with a larger network and our method does as well as PPO. As resources are limited in terms of the number of parameters and models become less complex, it seems natural that filtering samples according to their expected informational value helps to reduce noise in the gradient update and to speed up learning.

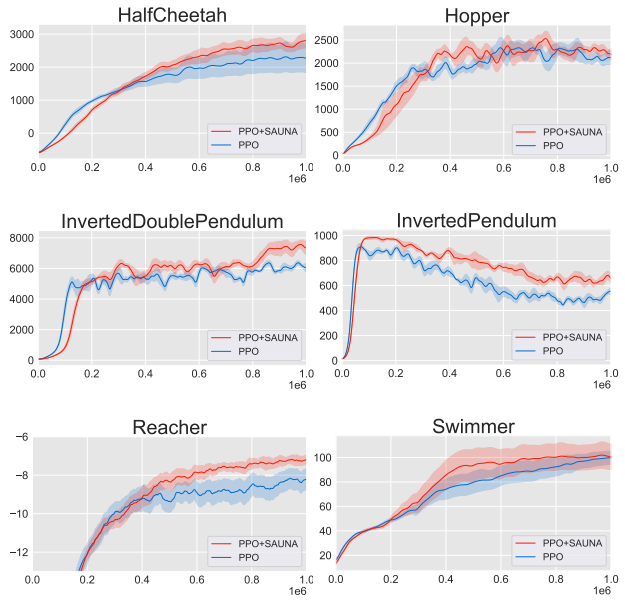
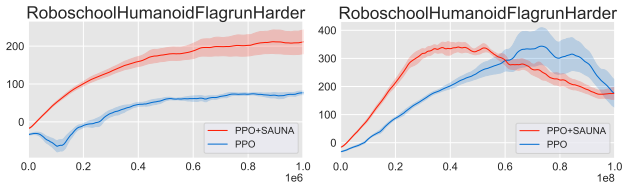


Figure 1: Performance of PPO+SAUNA (red) relative to PPO (blue) on 6 *MuJoCo* environments averaged across 6 seeds. X-axis: number of environment steps. Y-axis: total undiscounted return. Shaded areas: standard deviation.



(a) (b)

Figure 2: Performance of PPO+SAUNA (red) relative to PPO (blue) on the *Roboschool* environment averaged across 6 seeds. X-axis: number of environment steps. Y-axis: total undiscounted return. Shaded areas: standard deviation.

5.2 Learning with SAUNA

The Advantages of Filtering

We further study the impact of filtering out noisy samples by conducting additional experiments in predicting $\mathcal{V}^{ex}$ while omitting the filtering step: the *if* statement (Line 8 of Algorithm 1) is removed and all transitions are kept in the batch $\mathcal{B}$. Indeed, SAUNA may improve the agent's performance by simply training the shared network to optimize the $\mathcal{V}^{ex}$ head as an auxiliary task. Fig. 3 demonstrates the positive effects of filtering out the samples. In addition, we studied the number of filtered out samples per task and its evolution along the training. On average, SAUNA rejects 5-10% of samples at the beginning of training, 2-6% near the end.

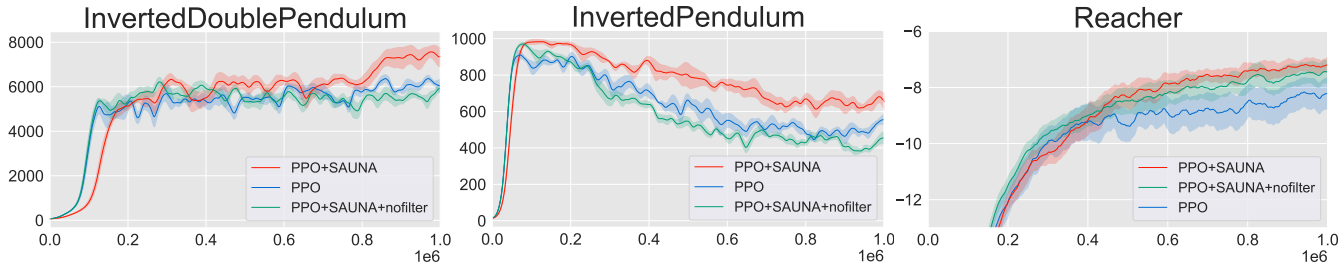
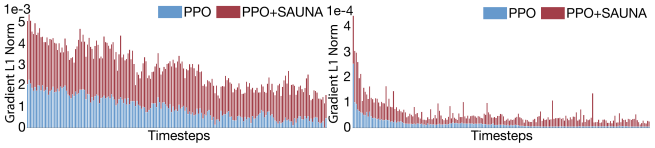


Figure 3: Performance of PPO+SAUNA (red) relative to PPO (blue) and PPO with the prediction of $\mathcal{V}^{ex}$ but without the filtering out of noisy samples (orange) on 3 *MuJoCo* environments averaged across 6 seeds. X-axis: number of environment steps. Y-axis: total undiscounted return. Shaded areas: standard deviation.

The Impact of SAUNA on the Gradients

Prior to the gradient update, SAUNA removes the useless transitions. By so doing, we hypothesized that information signals from samples with large $\mathcal{V}^{ex}$ would be less diluted by filtering out samples. Fig. 4 shows that SAUNA filtering



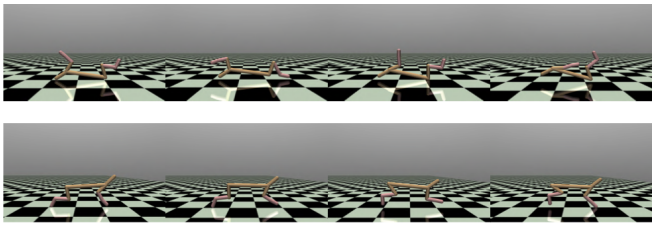
(a) (b)

Figure 4: Gradients L1-norm from the (a) first layer and (b) last layer of the shared parameters network for PPO and PPO coupled with SAUNA. Task: *HalfCheetah-v2*.

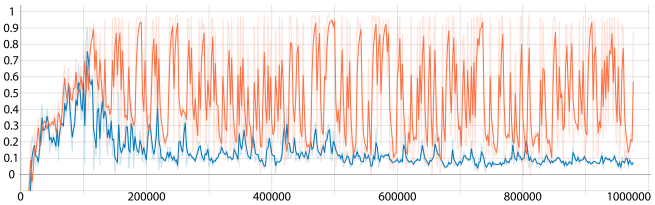
leads to larger gradients. As a result, policy updates make bigger steps, which ultimately translates into better performance. It is questionable why performance is not negatively affected, since larger gradients could hinder learning. Experience shows that gradients contain more useful information: as the relevant signals are less diluted, the gradients are more qualitative and have been partially denoised.

HalfCheetah: Qualitative Study

In *HalfCheetah*, a well-known behavior [Lapan, 2018] is that for multiple seeds PPO is stuck in a local minimum in which the agent moves on its back. However, we observed that SAUNA made it possible to leave from, or at least to avoid these local minima. This is illustrated in Fig. 5a where we see still frames of two agents trained with PPO+SAUNA for 10^6 timesteps on identically seeded environments. Their behavior is entirely different. Looking at $\mathcal{V}^{ex}$ in Fig. 5b, we can see that the graphs differ quite interestingly. The orange agent seems to find very quickly a local minimum on its back while the blue agent's $\mathcal{V}^{ex}$ varies much more. This seems to allow the latter to explore more states than the former and finally to find a better optimum. Supported by the previous study, we can infer that agents trained with SAUNA are better able to explore interesting states while exploiting with confidence the value given to the states observed so far.



(a)



(b)

Figure 5: (a) Example of PPO getting trapped in a local minimum (top row) while PPO+SAUNA reaches a better optimum (bottom row). (b) $\mathcal{V}^{ex}$ score for PPO (orange) and PPO+SAUNA (blue).

6 Discussion

Intuitively, for the policy update, our method will only use qualitative samples that provide the agent with (a) reliable and exercised behavior (high $\mathcal{V}^{ex}$) and (b) challenging states from the point of view of correctly predicting their value (low $\mathcal{V}^{ex}$). SAUNA algorithm keeps samples with high learning impact, rejecting other noisy samples from the gradient update.

6.1 Filtering Policy Gradient Updates and the Policy Gradient Theorem

Policy gradient algorithms are backed by the policy gradient theorem [Sutton *et al.*, 2000]. As long as the asymptotic stationary regime is not reached, it is not reasonable to assume the sampled states to be independent and identically distributed (i.i.d.). Therefore, it seems intuitively better to ignore some of the samples for a certain period, to allow the most efficient use of information. One can understand SAUNA as making gradient updates more robust through filtering, especially when the update is low and the noise can be dominant. Besides, filtering out disturbing samples reduces the bias in the state distribution.

6.2 Learning $\mathcal{V}^{ex}$ and the Shared Network Parameters

SAUNA network predicts $\mathcal{V}^{ex}$ in conjunction with the value function and the policy. Therefore, as its parameters are updated through gradient ascent, they converge to one of the objective function minima (hopefully, a global minimum). This parameter configuration integrates $\mathcal{V}^{ex}$, predicting how much the value function has fitted the observed samples, or informally speaking how well the value function is doing for state s_t. This new objective tends to lead the network to adjust predicting a quantity relevant for the task. Instead of using domain knowledge for the task, the method rather introduces problem knowledge by constraining the parameters directly.

6.3 Additional Experimental Results

We also compare SAUNA to A2C, a synchronous variant of [Mnih *et al.*, 2016] and a weaker version of PPO. As expected, we observe a 15% increase in performance. We do not present the complete results in this version of the article due to space limitations. Below is a discussion about additional experimental results, which we think contribute interestingly to the study.

Mean of $\mathcal{V}^{ex}$. Although $\widetilde{\mathcal{V}^{ex}}$, the median of $\mathcal{V}^{ex}$, is more expensive to calculate, we observe that it gives better results than if we use its mean in the *if* statement of Algorithm 1. Using the median helps [Kvålseth, 1985] because the distribution of $\mathcal{V}^{ex}$ is not normal and includes outliers that will potentially produce misleading results.

Non-empirical $\mathcal{V}^{ex}$. We also experimented with using the empirical values of $\mathcal{V}^{ex}$ in Line 8 of Algorithm 1 when calculating $\widetilde{\mathcal{V}^{ex}}$, instead of the predicted ones. This has yielded less positive results, and it is likely that this is due to the difference between the predicted and actual values at the beginning of learning, which has the effect of distorting the ratio in the *if* statement.

Adjusting state count. In order to stay in line with the policy gradient theorem [Sutton *et al.*, 2000], we have worked to adjust the distribution of states d^{π} to what it truly is, since some states visited by the agent are not included in the batch. We adjusted it using the ratio between the number of states visited and the actual number of transitions used in the gradient update, but this did not improve the learning, and instead, we observed a decrease in performance.

Adjusted $\mathcal{V}^{ex}$. The definition of $\mathcal{V}^{ex}$ is biased. An unbiased estimator does exist (known in statistics as the adjusted R^2). We performed the same set of experiments using such an adjusted $\mathcal{V}^{ex}$: it did not change the experimental performance significantly.

Random filtering. We experimented with dropping out at random, and before each gradient update, a number of samples corresponding to the same average number of samples that SAUNA drops. This resulted in a decrease in performance compared to PPO, as one can expect.

Atari domain. We tested our method on the *Atari 2600* domain [Bellemare *et al.*, 2013] without observing any improvement in learning: some of the tasks were best performed by one method and others by the other.

7 Conclusion

Policy gradient methods optimize the policy directly through gradient ascent. We have introduced a new, lightweight and agnostic method applicable to any policy gradient algorithm. The central idea of this paper is that $\mathcal{V}^{ex}$ is a useful measure to filter out samples that are perturbing the policy update. Those non-informative or misinformative samples are ignored by SAUNA with a mechanism controlled by the estimated fraction of variance explained by the value function at each state. The relevant signals being less diluted, this improved sampling results in a denoising effect on the gradients, improving the learning curve, ultimately leading to improved performance.

We demonstrated the effectiveness of our method when applied to PPO, a commonly used state of the art policy gradient method, on a set of benchmark high-dimensional environments. We also established that samples can be removed from the gradient update without hindering learning but, on the opposite, can improve it. We further studied the positive impacts that such a modification in the sampling procedure has on learning. Several open topics warrant future study. Our results suggest that the influence of SAUNA on the distribution of states has beneficial effects: in order to gauge the theoretical implications of transition dropout in the MDP, our method might be formulated using the options framework [Sutton *et al.*, 1999; Precup, 2000] where holes in a trajectory result in the appearance of options. Moreover, we are studying other ways to use $\mathcal{V}^{ex}$ in the context of RL.

References

[Amari, 1998] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[Andrew *et al.*, 1983] A. Andrew, R. Sutton, and C. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(5):834–846, 1983.

[Bellemare *et al.*, 2013] M. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *JAIR*, 47:253–279, 2013.

[Brockman *et al.*, 2016] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. *arXiv:1606.01540*, 2016.

[Espeholt *et al.*, 2018] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *Proc. ICML*, 2018.

[Flet-Berliac and Preux, 2019] Y. Flet-Berliac and P. Preux. MERL: Multi-Head Reinforcement Learning. In *Deep Reinforcement Learning Workshop, NeurIPS*, 2019.

[Freeman *et al.*, 2019] D. Freeman, L. Metz, and D. Ha. Learning to predict without looking ahead: World models without forward prediction. In *Proc. NeurIPS*, 2019.

[Ha and Schmidhuber, 2018] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In *Proc. NIPS*, 2018.

[Haarnoja *et al.*, 2018] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proc. ICML*, 2018.

[Heess *et al.*, 2015] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa. Learning continuous control policies by stochastic value gradients. In *Proc. NIPS*, 2015.

[Hill *et al.*, 2018] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu. Stable baselines, 2018.

[Kakade, 2002] S. Kakade. A natural policy gradient. In *Proc. NIPS*, 2002.

[Kakade, 2003] S. Kakade. *On the sample complexity of reinforcement learning*. PhD thesis, University of London, 2003.

[Klimov and Schulman, 2017] O. Klimov and J. Schulman. Roboschool, 2017.

[Kvålseth, 1985] T. Kvålseth. Cautionary Note about R^2. *The American Statistician*, 39(4):279–285, 1985.

[Lapan, 2018] M. Lapan. *Deep Reinforcement Learning Hands-On*. Packt Publishing, 2018.

[Levine and Abbeel, 2014] S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Proc. NIPS*, 2014.

[Lillicrap *et al.*, 2016] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *Proc. ICLR*, 2016.

[Mnih *et al.*, 2016] V. Mnih, A. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proc. ICML*, 2016.

[Nachum *et al.*, 2017] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *Proc. NIPS*, 2017.

[Peters and Schaal, 2008] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

[Precup, 2000] D. Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts, 2000.

[Puterman, 1994] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[Robinson and Fallside, 1989] A. Robinson and F. Fallside. Dynamic reinforcement driven error propagation networks with application to game playing. In *Conference of the Cognitive Science Society*, 1989.

[Schaul *et al.*, 2016] T. Schaul, J. Quan, I. Antonoglou, and D. Silver. Prioritized experience replay. In *Proc. ICLR*, 2016.

[Schmidhuber and Huber, 1991] J. Schmidhuber and R. Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(1/2):135–141, 1991.

[Schmidhuber, 1991] J. Schmidhuber. Curious model-building control systems. In *Proc. IEEE IJCNN*, 1991.

[Schulman *et al.*, 2016] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proc. ICLR*, 2016.

[Schulman *et al.*, 2017] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[Silver *et al.*, 2014] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *Proc. ICML*, 2014.

[Silver *et al.*, 2016] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484, 2016.

[Srivastava *et al.*, 2014] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

[Sutton and Barto, 2018] R. Sutton and A. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Sutton *et al.*, 1999] R. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[Sutton *et al.*, 2000] R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Proc. NIPS*, 2000.

[Todorov *et al.*, 2012] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *IEEE/RSJ ICIRS*, 2012.

[Wang *et al.*, 2017] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, and N. de Freitas. Sample efficient actor-critic with experience replay. In *Proc. ICLR*, 2017.

[Werbos, 1989] P. Werbos. Neural networks for control and system identification. In *IEEE CDC*, 1989.

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

[Wu *et al.*, 2017] Y. Wu, E. Mansimov, R. Grosse, S. Liao, and J. Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Proc. NIPS*, 2017.