Off-Policy Shaping Ensembles in Reinforcement Learning

Anna Harutyunyan and Tim Brys and Peter Vrancx and Ann Nowé\textsuperscript{1}

Abstract. In this work we propose learning an ensemble of policies related through potential-based shaping rewards via the off-policy Horde framework.

1 Introduction and background

Ensemble techniques are widespread in supervised learning, but their use in reinforcement learning (RL) [9] has been extremely sparse thus far. We seek to formulate a RL ensemble that is effective at improving learning speed, the bottleneck of RL. In the context of this goal, the ensemble needs to learn in parallel efficiently. Recently proposed Horde architecture [10] fills this bill, and is the first to also possess convergence guarantees in realistic setups. In contrast to the previous uses of Horde, we exploit its power for learning a single task faster. The policies in our ensemble are obtained through potential-based reward shaping (PBRS), each expressing different pieces of heuristic domain knowledge, and are then combined via a voting rule. Maintaining multiple shapings allows leveraging the strengths of different heuristics without having to design a complex shaping reward [2].

The scenario we consider is that of off-policy learning under fixed behavior, i.e. latent learning. Such is often the setup in applications where the environment samples are costly and a failure is highly penalized. To our knowledge, this is the first validation of PBRS effective in such a latent setting, where it does not actively guide exploration.

For omitted details in discussion and experiment setup, please see the full version of this paper [5]. For standard background on reinforcement learning, see Sutton and Barto [9]. We briefly give the ingredients of the described approach.

Reward shaping augments the true reward signal with an additional heuristic shaping reward, provided by the designer, Ng et al. [8] show that grounding the shaping rewards in state potentials is both necessary and sufficient for ensuring preservation of the (optimal) policies of the original task. PBRS maintains a potential function $\Phi : S \to \mathbb{R}$, and defines the auxiliary reward function $F$ as $F(s,a,s') = \gamma \Phi(s') - \Phi(s)$, where $\gamma$ is the usual discounting factor.

Horde Learning about a (target) policy that is different from the (behavior) one currently followed, is referred to as learning off-policy. Despite its versatility, off-policy learning suffers from convergence issues, when combined with function approximation

\textsuperscript{1} AI Lab, Vrije Universiteit Brussel, Belgium, email: \{anna.harutyunyan, timbrys, pvrancx, anowe\}@vub.ac.be

\textsuperscript{2} See the discussion on convergence in Section 6.1.2 of van Hasselt’s dissertation [11].
faster knowledge propagation through the temporal-difference updates, which we now observe decoupled from guidance of exploration. These effects of off-policy reward shaping may be of independent interest.

3 Experiments

We focus our attention to a classical benchmark domain of mountain car [9]. The task is to drive an underpowered car up a hill. The base reward is $-1$ for each step, except the final one. We define three intuitive shaping potentials:

Position Encourage progress to the right (the goal): $\Phi_1(x) = c_r \times x$.

Height Encourage higher positions: $\Phi_2(x) = c_h \times h$.

Speed Encourage higher speeds: $\Phi_3(x) = c_s \times |\dot{x}|^3$.\(^3\)

We let the behavior be uniform over all actions (discrete throttle of $-1, 0, or 1$), and deploy Horde to concurrently learn\(^4\) the base task, and the three policies shaped w.r.t. the above potentials. We devise the ensemble policy via rank voting [12]. The evaluation is done by interrupting the learning every 5 episodes, and executing each greedy policy once. No learning was allowed during evaluation.

The speed shaping turns out to be the most helpful universally. If this is the case, one would prefer to just use that shaping on its own, but we assume no access to such information a priori. To make our experiment more interesting, we consider two distinct scenarios: with and without this shaping. We would like our ensemble to outperform both comparable shapings in the former, and at least match the performance of the best one, in the latter case.

Table 1: Average of 1000 independent runs of 100 episodes each. Initial/final labels refer to the first/last 20% of a run. The results that are not significantly different from the best ($p > 0.05$) are in bold.

<table>
<thead>
<tr>
<th>Variant</th>
<th>Cumulative</th>
<th>Initial</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without best shaping</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No shaping</td>
<td>-386.3 ± 279.5</td>
<td>-868.7 ± 385.9</td>
<td>-185.1 ± 9.9</td>
</tr>
<tr>
<td>Right shaping</td>
<td>-310.4 ± 96.9</td>
<td>-378.5 ± 217.4</td>
<td>-290.3 ± 19.3</td>
</tr>
<tr>
<td>Height shaping</td>
<td>-283.2 ± 205.2</td>
<td>-594.2 ± 317.0</td>
<td>-182.3 ± 7.5</td>
</tr>
<tr>
<td>Ensemble</td>
<td>-211.2 ± 94.2</td>
<td>-330.6 ± 179.5</td>
<td>-180.2 ± 1.5</td>
</tr>
<tr>
<td>With best shaping</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No shaping</td>
<td>-391.7 ± 283.2</td>
<td>-813.5 ± 373.7</td>
<td>-193.2 ± 10.9</td>
</tr>
<tr>
<td>Right shaping</td>
<td>-303.4 ± 81.4</td>
<td>-346.7 ± 181.2</td>
<td>-295.1 ± 16.7</td>
</tr>
<tr>
<td>Height shaping</td>
<td>-292.4 ± 213.8</td>
<td>-619.8 ± 328.3</td>
<td>-190.1 ± 5.3</td>
</tr>
<tr>
<td>Speed shaping</td>
<td>-158.6 ± 23.7</td>
<td>-182.1 ± 50.6</td>
<td>-150.2 ± 2.9</td>
</tr>
<tr>
<td>Ensemble</td>
<td>-168.7 ± 44.7</td>
<td>-214.8 ± 94.8</td>
<td>-161.7 ± 4.0</td>
</tr>
</tbody>
</table>

The results in Table 1 show that individual shapings alone aid learning speed significantly, and the ensemble policy meets our desiderata: it either statistically matches or is better than the best shaping at any stage. The exception is the final performance in the second scenario, but the difference in the collected reward is still rather small.

4 Conclusions and future work

We believe the Horde architecture to be well-suited for ensemble learning in general, and, as it provides the necessary tools to learn many PBRS policies simultaneously at no added cost, a convenient framework for leveraging diverse heuristic knowledge. We demonstrated our method to be effective even on a simple task, and with an ad-hoc combination method. Larger problems with many locally good shapings are the target benchmark, and we expect them to yield larger benefits.

There are many directions for future work. Latent parallel learning of diverse value functions suggests exploring ways to learn good combination strategies, or the potential functions themselves. Naturally, such meta-learning has to happen at a much faster pace in order to be useful in speeding up the main learning process. The scalability of Horde allows for learning thousands of value functions efficiently. While it is rarely sensible to define thousands of distinct shapings, one could imagine maintaining many different scaling factors for the existing shaping potentials. This would not only mitigate the scaling problem, but make the representation more flexible by having non-static scaling factors throughout the state space.

The primary limitation of Horde is the requirement to keep the behavior policy fixed (or change it slowly). Relaxing this constraint is a topic of ongoing research. Horde tackles convergence, which is one of the two main theoretical challenges with off-policy learning under FA. The other has to do with the quality of solutions under off-policy sampling, which may, in general, fall far from optimum. Kolter gives a way of constraining the solution space, achieving stronger guarantees [6], but his algorithm is quadratic in complexity and not scalable.

ACKNOWLEDGEMENTS

Anna Harutyunyan is supported by the IWT-SBO project MIRAD (grant nr. 120057). Tim Brys is funded by a Ph.D grant of the Research Foundation-Flanders (FWO). The authors thank the anonymous reviewers who helped to improve the paper.

REFERENCES


\(^3\) Here $x = (x, \dot{x})$ is the state (position and velocity), $h$ is the height of the hill, and $c = (c_r, c_h, c_s)$ is a vector of tuned scaling constants.

\(^4\) We used $\gamma = 0.99$, $\lambda = 0.4$, $\alpha = 0.1$, $\beta = 0.0001$, and approximated the state space via 10 tilings of $10 \times 10$. 