Shortcut Actor-Critic

Jungwoo Park

1 Motivation

Offline reinforcement learning (RL) aims to train a policy on a pre-collected dataset without environment interaction. Recent generative approaches, such as diffusion models and flow matching, effectively model the complex, multi-modal action distributions in the dataset. However, they require iterative multi-step generation, and backpropagating through these steps makes training unstable.

Flow Q-Learning (FQL) addresses this by distilling a multi-step teacher flow policy into a one-step student policy. However, this distillation phase is computationally expensive because it requires the teacher to perform multi-step ODE solving (Euler's method) during training.

To reduce training time and complexity, I apply shortcut models which performs one-step generation without explicit distillation. Unlike FQL, which requires two separate policy networks, a single shortcut model can handle both one-step and multi-step generation. By avoiding iterative ODE solving and bypassing the lossy distillation process, **Shortcut Actor-Critic** offers greater training efficiency and potentially superior performance.

The implementation is available at: https://github.com/jungwoopark01/shortcut-actor-critic.

2 Related Works

Flow Q-Learning (FQL). Flow Q-Learning (FQL) [3] is an offline RL method comprising a Behavioral Cloning (BC) teacher flow policy, an one-step student policy, and a critic. The BC flow teacher policy learns to transform Gaussian noise into offline dataset actions via flow matching. A separate one-step student policy is then distilled from the BC flow policy to map Gaussian noise directly to actions. The critic evaluates Q-values of the one-step policy. The one-step policy is trained to fulfill two objectives: minimize the distillation loss by matching the BC flow policy's output and maximize the critic's Q-value. However, the distillation step requires iteratively solving the ODE for the BC flow policy, creating a significant computational burden.

Shortcut Models. Shortcut models [1] are generative models trained to produce high-quality images in a single step. While standard flow matching trajectories are curved, shortcut models learn a vector field conditioned on a time step size d to jump ahead. If d is small, the model predicts the tangent vector (instantaneous velocity) of the curved trajectory. If d is large, it predicts the secant line directly connecting the current point t to the destination at $t+d$. Instead of distillation, shortcut models use the self-consistency loss: one big jump with step size $2d$ from t to $t+2d$ should land in the same point as two smaller jumps of t to $t+d$ and $t+d$ to $t+2d$. This allows the model to learn large jumps without solving the full ODE trajectory.

3 Shortcut Actor-Critic

To eliminate the computational cost of ODE solving during training, I introduce **Shortcut Actor-Critic,** replacing the distillation loss of FQL with the self-consistency loss of a shortcut model. Algorithm 1 demonstrates the full algorithm of Shortcut Actor-Critic. The critic Q_ϕ is trained with the standard Bellman error,

using Polyak averaging for the target networks and Double Q-Learning to mitigate value overestimation problem. The shortcut model actor v_θ is trained using three loss components. The BC flow loss $\mathcal{L}_{\text{BC}}$ ensures the shortcut model to learn the action distribution at small step sizes by minimizing the standard flow matching loss. The self-consistency (SC) loss $\mathcal{L}_{\text{SC}}$ teaches the shortcut model to perform large jumps without solving the ODE.

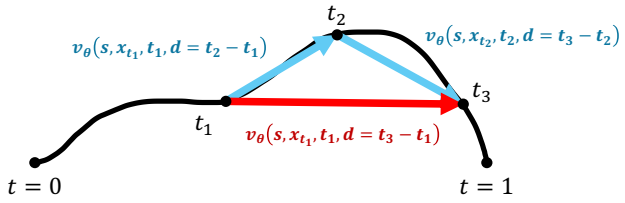

Figure 1: **Overview of self-consistency loss.** Shortcut models learns to jump a big step by matching to the destination of two small steps.

Figure 1 depicts how self-consistency loss is formulated geometrically. Q loss $\mathcal{L}_Q$ updates the actor to maximize the estimated Q-value, which is the average of the two Q networks. By jointly minimizing the BC and SC losses while maximizing the Q-value, the shortcut model actor learns a policy that is both consistent with the offline dataset and optimal with high Q-value actions.

4 Experiments

I conducted offline-to-online RL experiments to evaluate the performance and adaptability of Shortcut Actor-Critic compared to FQL on both settings. The evaluation was performed on two high-dimensional tasks from the OGBench [2]: `cube-double-play-singletask-v0` and `scene-play-singletask-v0`. To ensure statistical robustness, all results are averaged over 4 seeds. During the offline pre-training phase (0 to 0.5M steps), the policy is trained on the static offline dataset. The pre-trained policy interacts with the environment to improve itself during the online fine-tuning phase (0.5M to 1M steps).

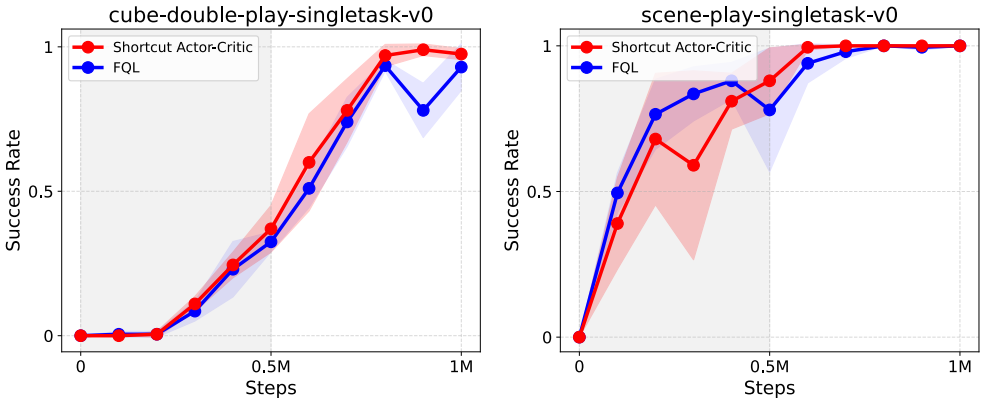

Figure 2: **Success rate comparison.**

Figure 2 presents the success rate comparison between Shortcut Actor-Critic and FQL. At the end of the offline phase, Shortcut Actor-Critic outperforms FQL on both tasks (0.37 vs. 0.325 and 0.88 vs. 0.78, respectively). Following the online fine-tuning phase, both models converge to a near-perfect success rate. To further distinguish the quality of the learned behaviors, I compared the average episode length in Figure 3.

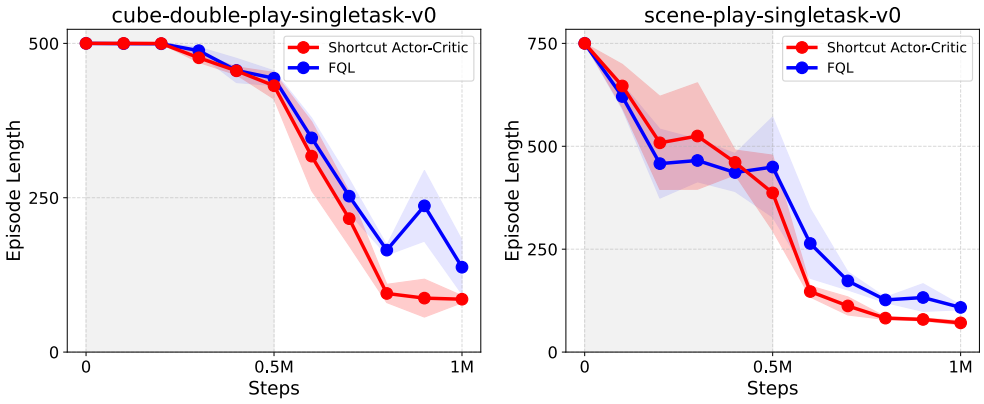

Figure 3: **Episode length comparison.**

A reduction in episode length indicates that the policy has improved upon the sub-optimal demonstrations in the dataset, identifying more efficient trajectories to complete the task. In both environments, Shortcut Actor-Critic significantly outperforms FQL with substantially shorter episodes (85.6 vs. 137.5 and 71.0 vs. 108.5). The simulation videos demonstrates the superior speed and efficiency of Shortcut Actor-Critic compared to FQL. The supplementary videos are available at Google Drive.

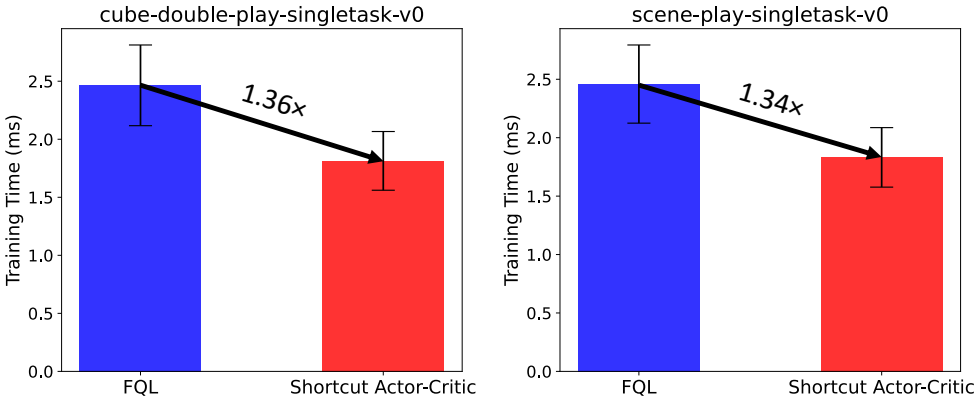

Figure 4: **Training time comparison.**

Finally, I evaluated the computational efficiency, which was the primary motivation for employing Shortcut Models. Figure 4 reports the training time for one offline pre-training step. Shortcut Actor-Critic achieves faster training time of 1.36× and 1.34× faster compared to FQL, validating the efficiency gains of avoiding multi-step ODE solving in FQL distillation.

5 Conclusion

In this work, I proposed Shortcut Actor-Critic, an RL algorithm designed to overcome the computational bottleneck of Flow Q-Learning (FQL). While flow matching offer powerful expressivness for multi-modal action distributions, it requires iterative ODE solving and two separate teacher and student models, which hinder training efficiency.

By integrating shortcut models, which learns to jump across the ODE trajectory curve, the need for the heavy distillation phase was successfully eliminated. Shortcut models allow the policy to learn one-step

action generation, while preserving the quality of flow-based modeling and reducing training overhead.

The experiments on high-dimensional OGBench tasks demonstrate that Shortcut Actor-Critic outperforms FQL in both offline pre-training and online fine-tuning. Shortcut Actor-Critic learns more efficient policies, as evidenced by reduced episode lengths. Furthermore, Shortcut Actor-Critic offers faster training phase. These results suggest that shortcut models are promising generative models for RL.

References

[1] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models, 2025.

[2] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl, 2025.

[3] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning, 2025.

Appendix

Algorithm 1 Shortcut Actor-Critic

Initialize θ, ϕ, $\phi_{\text{target}} = \phi$
while not converged **do**
Sample batch $\{(s, a, r, s')\} \sim \mathcal{D}$

▷ Train critic Q_ϕ
$z \sim \mathcal{N}(0, I_d)$
$a' \leftarrow z + \texttt{stopgrad}(v_\theta(s = s', x_t = z, t = 0, d = 1))$
$\boldsymbol{\mathcal{L}_{\text{critic}}(\phi) = \mathbb{E}\left[\left(Q_\phi(s, a) - r - \gamma Q_{\phi_{\text{target}}}(s', a')\right)^2\right]}$
$\phi_{\text{target}} \leftarrow \tau\phi + (1 - \tau)\phi_{\text{target}}$

▷ Train shortcut vector field v_θ
$x_0 \sim \mathcal{N}(0, I_d)$
$x_1 \leftarrow a$
$t \sim \text{Uniform}([0, 1])$
$x_t \leftarrow (1 - t)x_0 + tx_1$
$v^{\text{BC}}_{\text{target}} \leftarrow x_1 - x_0$
$\boldsymbol{\mathcal{L}_{\text{BC}} = \mathbb{E}\left[\left\|v_\theta(s = s, x_t = x_t, t = t, d = 1/2^M) - v^{\text{BC}}_{\text{target}}\right\|^2\right]}$
$m \sim \text{Uniform}(\{0, 1, \ldots, M - 1\}), d \leftarrow 1/2^{m+1}$
$u \sim \text{Uniform}(\{0, 1, \ldots, 2^m - 1\}), t \leftarrow u/2^m$
$x_t \leftarrow (1 - t)x_0 + tx_1$
$v_t \leftarrow v_\theta(s = s, x_t = x_t, t = t, d = d)$
$x_{t+d} \leftarrow x_t + v_t d$
$v_{t+d} \leftarrow v_\theta(s = s, x_t = x_{t+d}, t = t + d, d = d)$
$x_{t+2d} \leftarrow x_{t+d} + v_{t+d} d$
$v^{\text{SC}}_{\text{target}} \leftarrow \texttt{stopgrad}((x_{t+2d} - x_t)/2d) = \texttt{stopgrad}((v_t + v_{t+d})/2)$
$\boldsymbol{\mathcal{L}_{\text{SC}} = \mathbb{E}\left[\left\|v_\theta(s = s, x_t = x_t, t = t, d = 2d) - v^{\text{SC}}_{\text{target}}\right\|^2\right]}$
$z \sim \mathcal{N}(0, I_d)$
$a_\pi \leftarrow z + v_\theta(s = s, x_t = z, t = 0, d = 1)$
$\boldsymbol{\mathcal{L}_Q = -\mathbb{E}[Q_\phi(s, a_\pi)]}$

$\boldsymbol{\mathcal{L}_{\text{actor}}(\theta) = \alpha_{\text{BC}}\mathcal{L}_{\text{BC}} + \alpha_{\text{SC}}\mathcal{L}_{\text{SC}} + \mathcal{L}_Q}$
end while
return $v_\theta(s, x_t, t, d), Q_\phi(s, a)$