computer sciences and mathematics forum

MDPI

Proceeding Paper

Small Dataset, Big Gains: Enhancing Reinforcement Learning by Offline Pre-Training with Model-Based Augmentation †

Girolamo Macaluso [1,*], Alessandro Sestini [2] and Andrew D. Bagdanov [1]

[1] Media Integration and Communication Center, University of Florence, 50134 Florence, Italy; andrew.bagdanov@unifi.it
[2] SEED - Electronic Arts, Södermalmsallén 36, 118 28 Stockholm, Sweden; asestini@ea.com
* Correspondence: girolamo.macaluso@unifi.it
† Presented at the 2nd AAAI Workshop on Artificial Intelligence with Biased or Scarce Data (AIBSD), Vancouver, BC, Canada, 26 February 2024.

Abstract: Offline reinforcement learning leverages pre-collected datasets of transitions to train policies. It can serve as an effective initialization for online algorithms, enhancing sample efficiency and speeding up convergence. However, when such datasets are limited in size and quality, offline pre-training can produce sub-optimal policies and lead to a degraded online reinforcement learning performance. In this paper, we propose a model-based data augmentation strategy to maximize the benefits of offline reinforcement learning pre-training and reduce the scale of data needed to be effective. Our approach leverages a world model of the environment trained on the offline dataset to augment states during offline pre-training. We evaluate our approach on a variety of MuJoCo robotic tasks, and our results show that it can jumpstart online fine-tuning and substantially reduce—in some cases by an order of magnitude—the required number of environment interactions.

Keywords: offline reinforcement learning; world models; data augmentation; fine-tuning

Citation: Macaluso, G.; Sestini, A.; Bagdanov, A.D. Small Dataset, Big Gains: Enhancing Reinforcement Learning by Offline Pre-Training with Model-Based Augmentation. *Comput. Sci. Math. Forum* **2024**, *9*, 4. https://doi.org/10.3390/cmsf2024009004

Academic Editors: Kuan-Chuan Peng, Abhishek Aich and Ziyan Wu

Published: 18 February 2024

1. Introduction

Effective policy learning with reinforcement learning (RL) often demands extensive interaction with the environment, a process that can be both expensive and potentially unsafe in real-world scenarios, such as robotics, logistics, and autonomous driving, where exploration with untrained policies is either costly or poses safety risks [1]. In some cases, pre-collected experience datasets are available, offering a valuable resource.

Offline RL has emerged as a technique to leverage such dataset to train effective policies, eliminating the need for continuous interaction with the environment during training. In offline RL, no assumptions are made about the policy used to gather the data, known as the behavioral policy. The primary objective of offline RL is, indeed, to create a policy that outperforms the performance of the behavioral policy. However, this poses significant challenges. Enhancing the policy beyond the performance of the behavior policy that gathered the data involves estimating values for actions not encountered in the dataset. This introduces the risk of distributional shift, where the policy may face states or actions during deployment that differ significantly from those in the training data. To reduce this problem, several offline RL approaches introduce constraints to the learned policy [1].

A promising approach to reduce the sample complexity and training time of RL is taking advantage of pre-collected experience combining offline and online RL [2,3]: the first is used to create an initialization that jumpstarts the online training. However, we find that the size of the offline dataset significantly influences the effectiveness of offline pre-training, sometimes even slowing down the online process significantly. This is due to the overfitting that happens when combining offline RL and small datasets.

In this work, we introduce a model-based approach to enhance the offline RL pre-training stage (see Figure 1) that fully leverages the potential of the offline dataset to speed

up online training and significantly reduce the number of online environment interactions needed. Our proposed approach is based on a generative world model, trained end-to-end by using the offline dataset, that can generate the next state given the actual state and the action. This capability is exploited during offline training to augment the transition used, creating a policy that is more informed about the environment. We conduct experimentation on a very small sample of the D4RL [4] offline MuJoCo [5] datasets to simulate a scarce data scenario. We also try to understand what is the impact of our augmentation on the learned initialization. We use the Twin Delayed Deep Deterministic Policy Gradient (TD3) for the online fine-tuning and its offline version Twin Delayed Deep Deterministic Policy Gradient with Behavioral Cloning (TD3BC) for the pre-training.

Our primary contribution is a framework designed to train policies, maximizing the information gained from small pre-collected experience datasets and enhancing the sample efficiency. We begin by training a generative world model by using the offline dataset, and subsequently, we leverage this model to augment the offline pre-training, enhancing the quality of the online initialization. This effectively accelerates the fine-tuning process, enabling the policy to achieve the same return value as a fully online-trained policy in significantly fewer iterations, sometimes even by an order of magnitude.

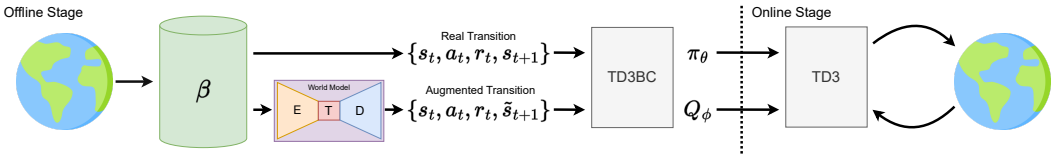


Figure 1. Our proposed training process: The offline dataset is used to train our world model (purple). We then use this model to augment half of the transitions sampled from the offline dataset β in a batch of TD3BC training samples. The resulting actor π_θ and critic Q_ϕ are then used as initialization for the online learning phase with TD3.

2. Related Work

In offline reinforcement learning (RL), we optimize a policy without relying on interactions with the environment but rather by using a fixed dataset pre-collected from a behavioral policy [6,7]. The task of offline RL is to train a policy that surpasses the performance of the behavioral policy, "stiching" together parts of different trajectories that contain good behaviors. This framework has recently gained interest within the research community for its potential in scenarios where extensive interactions with the environment are costly or unsafe, such as in real-world applications like robotics, logistics, and autonomous driving [1]. Another interesting use case of offline RL is to initialize an online RL training by using a dataset of experiences. In this section, we review work from the literature most related to our contributions.

2.1. Combining Offline and Online RL

The combination of offline and online RL techniques has emerged as a promising research direction. In works like [2,3,8], offline RL has been used to train a policy from a pre-collected dataset of experiences that is then fine-tuned with online RL. These studies have investigated diverse strategies aimed at improving the performance gain of offline pre-training and mitigating the phenomenon known as *policy collapse*, which causes a performance dip when shifting from offline to online training [3]. These studies reveal that the dip may be caused by the initial overestimation of the offline trained critic for states–action pairs unseen during training.

These approaches propose measures such as reducing the underestimation during offline stages [8], imposing a conservative improvement to the online stage [3], and weighting policy improvement with the advantage function [2].

Our work extends this research to scenarios with a severely limited offline dataset, introducing a model-based augmentation technique that enhances offline pre-training, taking full advantage of the available data.

2.2. World Models

World models (WMs) have attracted great attention in prior RL research, demonstrating their capacity to capture environmental dynamics and elevate the performance of agents. A WM trained through random exploration has exhibited strong performance when used as a proxy for policy learning in simple environments [9]. In such cases, the WM excels at predicting future outcomes, effectively simulating the true environment. The "*Dreamer*" framework [10,11] takes a more comprehensive approach, iteratively refining the WM. An agent is trained by using the WM "*imagined*" trajectories and is then deployed to gather new experiences, which are used to refine the WM. Remarkably, this framework has demonstrated considerable success, even learning a policy from scratch to obtain diamonds in the popular video game *Minecraft*, a very difficult task due to the gigantic state space and the sparsity of the reward [12]. Our approach builds on this concept of integrating a WM into the training process, but instead of simulating trajectories, we use the WM to augment the small available dataset during the offline training.

3. Effective Offline Pre-Training with Scarce Data

Our method centers around the development of a generative WM utilizing the offline dataset to enhance offline RL pre-training. The primary objective is to generate a more informed actor–critic pair that serves as a superior initialization for subsequent online training and that also diminishes the need for extensive online interactions, resulting in a more efficient and effective RL training paradigm.

3.1. World Model

The core of our approach is the generative WM designed as a variational autoencoder (VAE) with a transition model. This WM is responsible for encapsulating the environmental dynamics and providing the capability to predict the next state given the current state and action. The WM is composed of a variational encoder, a decoder, and the transition model.

3.1.1. Encoder–Decoder

In our VAE setup, the encoder projects the environment state s_t into a lower-dimensional latent space, parameterizing the mean μ_{s_t} and log variance $\log \sigma^2_{s_t}$ of a Gaussian distribution within this space. The current state's latent representation z_t is sampled from this distribution, introducing crucial stochasticity for capturing diverse environment dynamics. Our experiments will demonstrate that this stochasticity can prove beneficial, even in environments that are deterministic. This sampling allows our WM to generate different possible next states for the same state–action pair. Conversely, the decoder reconstructs the latent state back into the original state space. In our approach, the decoder is employed to project the next latent space $\widetilde{z}_{t+1}$, produced by the transition model, to the state space $\widetilde{s}_{t+1}$.

3.1.2. Transition Model

The transition model plays the role of predicting the changes that occur within the latent space due to the execution of an action. The ability to model these changes allows our system to predict future states and thus simulate forward in time. The transition model takes as input the latent representation of the current state z_t and the action a_t to be performed. Utilizing this information, the transition model predicts the change $\Delta(s_t, a_t)$ in the latent space that would occur due to the action a_t. This is then added back to the current latent state representation z_t to obtain the next latent state $\widetilde{z}_{t+1}$, represented as $\widetilde{z}_{t+1} = z_t + \Delta(s_t, a_t)$. This provides an estimate of the subsequent state's latent representation in the environment given the current state and action. We designed the transition model to predict the changes within the latent space rather than modeling the next latent space directly, making it a model of the variation induced by an action within a specified latent space.

3.1.3. Training Procedure

Our WM is trained end-to-end on the offline dataset, with the objective of enabling the model to generate the next state given the current state and action. Our model optimizes the Evidence Lower Bound (ELBO) loss along with two additional reconstruction losses, illustrated in Figure 2:

$$\mathcal{L}_{\text{wm}} = \underbrace{\text{MSE}(\widetilde{s}_t, s_t) + \text{KL}(\mathcal{N}(0, I) \,||\, \mathcal{N}(\mu_{s_t}, \sigma^2_{s_t}))}_{\mathcal{L}_{\text{ELBO}}} + \underbrace{\text{MSE}(\widetilde{s}_{t+1}, s_{t+1})}_{\mathcal{L}_s} + \underbrace{\text{MSE}(\widetilde{z}_{t+1}, z_{t+1})}_{\mathcal{L}_z}.$$

The ELBO loss, a standard component in VAEs, plays a pivotal role in ensuring the meaningfulness of the learned latent representation. Its contribution lies in shaping a compact and dense latent space. Given that our WM optimizes this loss, it can effectively function as a variational autoencoder when utilizing only the encoder and decoder components. The training of the transition model incorporates two reconstruction losses: one for the latent space $\mathcal{L}_z$ and another for the state space $\mathcal{L}_s$. The latent space loss aims to maintain coherence between the predicted latent representation and the one sampled by the encoder by using the actual next state. Simultaneously, the state space loss ensures the model's ability to reconstruct the next state within the state space. Both reconstruction losses are calculated by using the Mean Squared Error (MSE) metric.

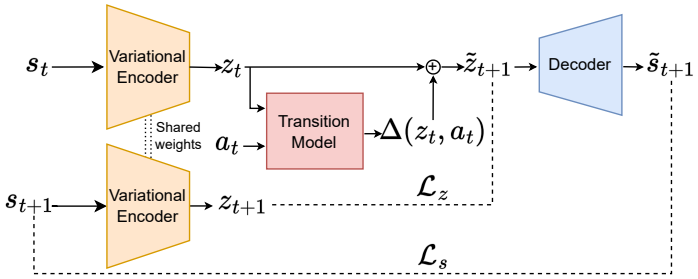


Figure 2. Our world model training procedure. The encoder (orange) takes the state s_t and outputs mean and variance of a Gaussian used to sample the latent representation z_t. Then, the transition model (red) takes z_t and action a_t and generates the change in the latent space $\Delta(s_t, a_t)$ caused by the action a_t in the latent state z_t. This is added back to the latent state to reconstruct the latent representation of the next state $\widetilde{z}_{t+1}$. The decoder (blue) then reconstructs the next state $\widetilde{s}_{t+1}$ starting from $\widetilde{z}_{t+1}$. The real next state s_{t+1} is forwarded into the variational encoder obtaining z_{t+1} that is used as target in the latent space reconstruction loss $\mathcal{L}_z$. The state space reconstruction loss in the state space $\mathcal{L}_s$ is instead computed between s_{t+1} and $\widetilde{s}_{t+1}$.

3.2. Offline Pre-Training

We perform the offline training with Twin Delayed Deep Deterministic Policy Gradient with Behavior Cloning (TD3BC) [13], an actor–critic off-policy offline RL algorithm. During the training, a part of the transition within the batch sampled from the offline dataset is augmented with the use of our WM. The augmentation consists of substituting the next state of 50% of the transitions sampled in a training batch of TD3BC with one generated by our WM. The sampled transition tuple (s_t, a_t, r_t, s_{t+1}) becomes $(s_t, a_t, r_t, \widetilde{s}_{t+1})$, with $\widetilde{s}_{t+1}$ being the state predicted by the WM by using the current state s_t and the action a_t. Augmented next states are not stored; each time a transition is sampled from the replay buffer, it can be dynamically augmented with a newly generated next state.

Our augmentation technique has an impact on the computation of target Q-values, which plays a central role in the temporal difference learning of the critic of TD3BC [14]. In the conventional approach, the target Q-values rely exclusively on the next states present in the offline dataset. This can lead to the overfitting of the Q-function if the dataset is particularly small. This overfitting manifests itself as narrow peaks in the state–action value space, each corresponding to states observed in the dataset. Overfitting the Q-function can

be detrimental to the overall performance of the offline RL algorithm as it may struggle to generalize effectively to unseen or underrepresented states in the environment.

By introducing the augmented next state generated by our WM into the computation of target Q-values, we aim to mitigate this risk and smooth out the narrow peaks by enriching the state–action value space with a more diverse set of states. This regularizing effect promotes a broader and more robust representation of the environment dynamics and makes the TD3BC algorithm less sensitive to parts of the environment that may not be adequately represented in the offline dataset.

Intuitively, this augmentation strategy serves as a guided exploration of the state space that aligns with the WM's interpretation of the environmental transition dynamics. It leverages the WM's predictive capabilities to envision different potential future outcomes and effectively expand the states considered during offline training.

We decided to employ our WM for generating predictions only one step into the future. This cautious decision is in line with the constrained size of the dataset, intending to find a balance that utilizes the WM's predictive capabilities while minimizing the risk of accumulating errors caused by data scarcity.

3.3. Online Fine-Tuning

Our offline trained actor and critic serve as initialization for the online RL phase with Twin Delayed Deep Deterministic Policy Gradient (TD3) [14]. We maintain the TD3BC framework's setup for the offline stage, specifically retaining state normalization. Upon transitioning to TD3, we continue normalization to effectively leverage the actor and critic. To perform the normalization, we utilize the statistics from the offline dataset. The transition from TD3BC to TD3 is straightforward, involving the removal of the behavioral cloning term from the actor loss and the initialization of the online replay buffer with transitions sampled by using the actions chosen by the pre-trained actor.

The offline initialization aims to accelerate training by reducing the need for exhaustive online interaction with the environment. Offline training with data augmentation leads to a more informed actor–critic pair that jumpstarts the online training by offering more reliable insight into state–action pairs. In the following section, we present an ablation study investigating the influence of the offline initialization of only actors or critics on the online training performance, aiming to evaluate the individual roles of each component.

4. Experimental Results

In this section, we report on experiments that demonstrate the ability of our approach to perform data augmentation and improve pre-training, which in turn enables more effective and efficient online fine-tuning.

4.1. Experiments

The architecture of our encoder, decoder, and transition model is a four-layer Multi-Layer Perceptron (MLP), with a hidden layer of size 512 and Rectified Linear Unit (ReLU) activations. The architecture of both the policy and critic is the same as those used in the original TD3BC paper [13], a three-layer MLP with ReLU activations and a hidden layer size of 256.

We conducted experiments on four MuJoCo locomotion tasks: ant, hopper, walker, and halfcheetah [5]. Our experiments will demonstrate that our approach is beneficial, even in these deterministic environments. For each task we used the *medium-expert*, *medium-replay*, and *random* datasets from D4RL [4]. We sampled 10,000 transitions from each of these datasets, which represent only a tiny fraction of the original 2 million transitions for *medium-expert*, 200,000 for *medium-replay*, and 1 million for *random*. These samples of the original transitions are used as our offline datasets in order to simulate a scarce data scenario. These datasets are leveraged to train our WM and for the offline RL training with TD3BC.

The offline stage is performed by using our data augmentations for 200,000 iterations. The final online fine-tuning stage, initialized with the offline trained actor and critic, is then performed for another 500,000 iterations.

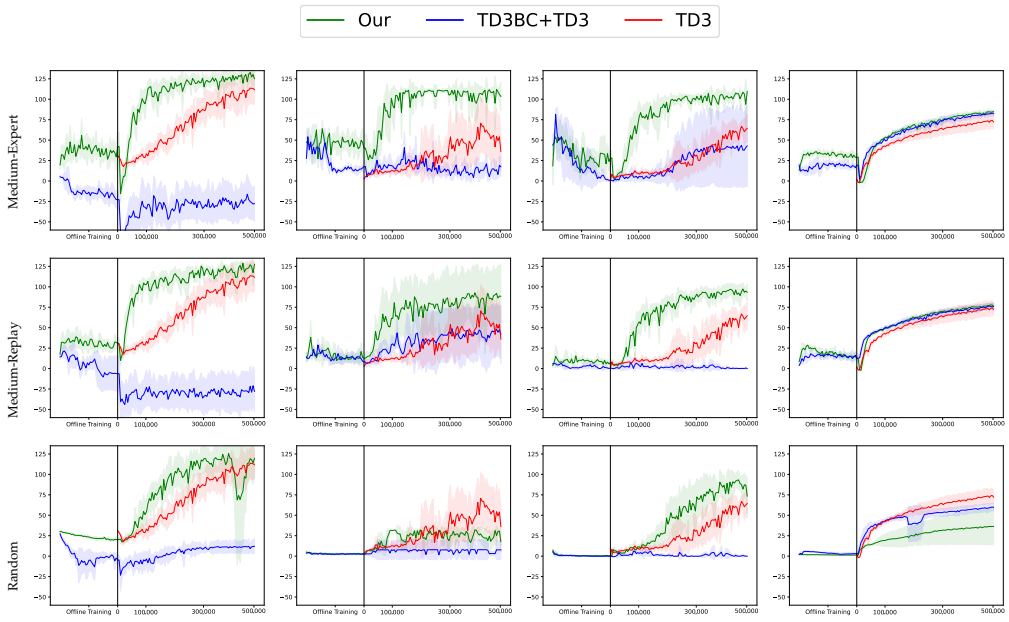
We compare our approach with two baselines:

- **TD3BC+TD3**, in which we use vanilla TD3BC for offline training and then use the learned policy and critic to initialize online learning with TD3 (initializing the replay buffer by using the actions from the actor, as in our approach).
- **Fully Online TD3**, in which we train the policy and the critic from scratch by using the off-policy online RL algorithm TD3.

The results are evaluated in terms of the normalized score [4], which is zero when the evaluated policy has the performance of a random policy, 100 when it has the same performance of a policy trained online with the Soft Actor–Critic (SAC) algorithm [15], and over 100 if it surpasses that performance. The evaluation is performed every 5000 training iterations. We also show an ablation study on the role of the actor and the critic in jumpstarting the online training and an intuition of why our augmentation is effective.

4.2. Comparative Performance Evaluation

Figure 3 illustrates the results of our experiments on all environments. We observe a consistent pattern across nearly all configurations; when transitioning from offline to online, there is evidence of *policy collapse* in which the performance drops significantly [3]. As observed by Luo et al. [3], when the data in an offline dataset are more diverse, e.g., in the *medium-replay* and *random* datasets, this performance dip is reduced.



(**a**) Ant (**b**) Hopper (**c**) Walker (**d**) Halfcheetah

Figure 3. Performance comparison between our approach (green line), TD3BC followed by TD3 (blue line), and full online training with TD3 (red line), measured in terms of normalized score. The segment preceding the black vertical line represents the scores achieved during offline training, while the subsequent segment reflects the transition to online training.

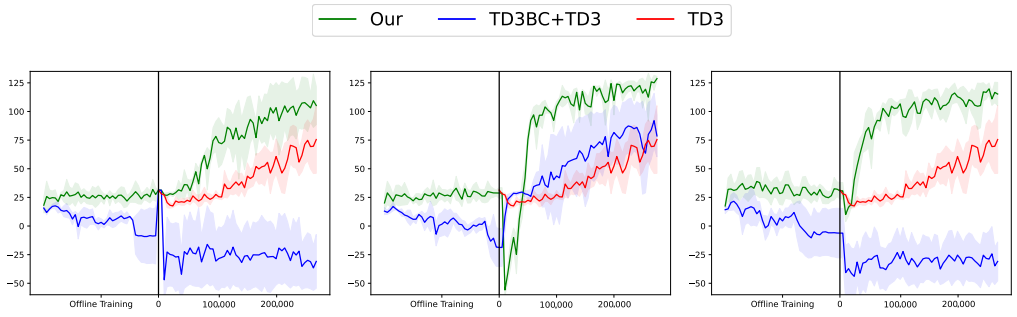
These results show that our augmented offline initialization significantly improves the online training performance after the initial dip. This is due to the more informed actor–critic pair that better mitigates overestimation when encountering states not seen during offline training. This improvement in learning speed reduces the number of online interactions needed to learn a performant policy, as well as reducing the sample complexity and the risks linked to initial exploration in real-world environments.

Figure 3 also illustrates the potential drawbacks of combining offline and online RL with limited data when not utilizing our augmentation. In all datasets and environments, employing vanilla TD3BC for policy initialization, in the best-case scenario, maintains a similar performance to fully online training. However, it can also lead to a complete failure of the online training, showcasing the challenges associated with scarce data. This becomes evident across multiple configurations, including those from the ant environment, as well as in walker with the *random* and *medium-replay* datasets and hopper with the *random* and *medium-expert* datasets.

The performance of our technique is linked to the quality of the dataset available: when a dataset with expert or medium transitions is available, the performance improves more consistently. Note how in the *ant*, *hopper*, and *walker* environments we are able to achieve the same performance of SAC (i.e., a normalized score of 100) by using only 100,000 online iterations when at least *medium* quality transitions are included in the dataset. When the dataset, in addition to being small, is composed of randomly collected transitions, the task becomes challenging. However, we still note a small improvement in the *ant* and *walker* environments. In the *halfcheetah* environment, however, the use of our augmentation does not provide any notable improvement, and is even *detrimental* when combined with a *random* dataset.

4.3. Ablations

We performed ablations on the offline initialization of the actor and critic to investigate if and how both contribute to the improved online learning performance observed in Figure 3. In Figure 4, we give the comparison of using both the offline pre-trained actor and critic on the online initialization in Figure 4c, only the critic in Figure 4b, and only the actor in Figure 4a. This experiment was conducted on the *ant medium-replay* dataset by using an offline dataset consisting of only 10,000 transitions.



(**a**) Pre-training only actor. (**b**) Pre-training only critic. (**c**) Pre-training actor and critic.

Figure 4. Ablation on offline pre-training of online actor and critic. These results demonstrate that pre-training of both actor and critic contribute to the accelerated online learning.

Using only the offline trained actor still yields a consistent boost in performance compared to the baselines. It is noteworthy that in this scenario, there is no performance dip. This is because policy collapse is linked to an initial overestimation of the critic when it encounters state–action pairs out of the training distribution [3]. This type of initialization may be useful in contexts in which it is unsafe to have a sharp decrease in performance.

On the other hand, when initializing the critic, we observe a substantial improvement following a significant dip in performance. This could be attributed to the inadequate initialization of the replay buffer, stemming from the use of a random actor during the initial online exploration. Additionally, the absence of the actor's balancing effect exacerbates the initial overestimation, contributing to the observed dip in performance. When considering the training using both the actor and critic, we observe a combination of the two behaviors, with better performance than using only the actor with a much smaller dip.

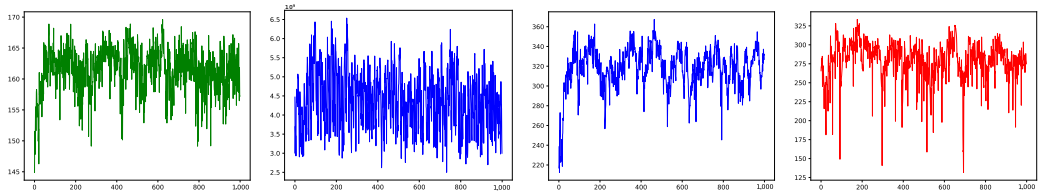
Without our data augmentation, using only the offline trained actor is detrimental to the training process and causes online training to fail completely. This highlights the critical

role of the pre-trained actor when our augmentation is not applied. In such cases, the quality of the initialization becomes pivotal, as a poor starting actor can lead to a complete failure in the online fine-tuning, even if a pre-trained critic is used. On the other hand, not initializing the actor helps to avoid the failure of the online fine-tuning. However, the offline pre-training is still less beneficial than using our augmentation, as the results are marginally above the one with both the policy and actor randomly initialized, indicated in red.

4.4. Why Pre-Training with Augmentation Is Effective

To gain deeper insight into the impact of data augmentation with our generative WM, we conducted a comparative analysis of the critic value. Specifically, we compare the critic value after training with our augmentation by using 10,000 transitions from the ant *medium-replay* dataset. This evaluation is compared against a critic trained by using 10,000 transitions on the same task without augmentation, a critic trained with the same setup as the latter but with 50,000 transitions, and a critic trained only online for 1 million iterations.

In Figure 5, we plot the results of this evaluation. In these plots, we show the value of the critic on the state–action pairs from the same episode sampled randomly from the *expert* D4RL dataset. We see in Figure 5b that the value of the critic trained without augmentation on 10,000 transitions falls drastically out of the environment's reward scale. This is likely due to the overfitting of the critic on the small offline dataset. However, in Figure 5c, as we increase the dimension of the offline dataset, we observe a more aligned critic resembling the one learned during the online stage, shown in Figure 5d.



(**a**) 10K transitions w/ aug. (**b**) 10K transitions w/o aug. (**c**) 50K transitions w/o aug. (**d**) Online trained policy.

Figure 5. Q-values of critic networks over the same single episode for (**a**) our approach with 10,000 transitions, (**b**) pre-training on 10,000 transitions *without* augmentations, (**c**) pre-training with TD3BC on 50,000 transitions *without* augmentations, and (**d**) online training with TD3.

In contrast, our augmentation yields a more conservative critic that closely aligns with the distribution of the online trained critic, as illustrated in Figure 5a. This suggests that our approach helps in maintaining a more reliable and cautious critic estimation, avoiding extreme values that could lead to overestimation issues. We believe that our augmentation process, involving the substitution of the evaluation state for the target critic during temporal difference learning with one generated by our WM, is simulating more exploration of the environment via the WM. This mechanism appears to contribute to a reduction in the overfitting.

5. Conclusions

In this paper, we proposed an approach to effectively leverage small datasets to reduce the sample complexity of reinforcement learning through offline reinforcement learning initialization. Our approach is based on a generative world model, trained on the offline dataset, that is able to predict state transitions. We propose an augmentation performed during the offline training based on the world model. Our experimental results show that conventional offline-to-online training, with limited datasets and without our augmentation, yields ineffective or even detrimental results. On the other hand, our approach offers a solution that maximizes the utility of the small offline dataset, successfully training a meaningful initialization that is able to speed up the online training. This approach

holds promise for improving the practical applicability of reinforcement learning in data-limited scenarios.

As future and ongoing work, we are adapting this technique to more challenging environments like the adroit manipulation tasks [16] in order to better assess the impact of data augmentation. We are also exploring the applicability and the impact of this technique on different offline and online reinforcement learning algorithms.

Author Contributions: G.M. is the primary author and made substantial contributions to the conceptualization, theoretical analysis, evaluations, and overall composition of the paper. A.S. and A.D.B. served as advisors, playing roles in validating the idea and experiments, reviewing and revising the manuscript, and providing indispensable support throughout the research process. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by European Commission Horizon 2020 grant #951911 (AI4Media).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The D4RL datasets are publicly available at https://github.com/Farama-Foundation/D4RL.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Levine, S.; Kumar, A.; Tucker, G.; Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv* **2020**, arXiv:2005.01643.
2. Nair, A.; Gupta, A.; Dalal, M.; Levine, S. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv* **2020**, arXiv:2006.09359.
3. Luo, Y.; Kay, J.; Grefenstette, E.; Deisenroth, M.P. Finetuning from Offline Reinforcement Learning: Challenges, Trade-offs and Practical Solutions. *arXiv* **2023**, arXiv:2303.17396.
4. Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; Levine, S. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *arXiv* **2000**, arXiv:2004.07219.
5. Todorov, E.; Erez, T.; Tassa, Y. Mujoco: A physics engine for model-based control. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Algarve, Portugal, 7–12 October 2012; pp. 5026–5033.
6. Lange, S.; Gabel, T.; Riedmiller, M. Batch reinforcement learning. In *Reinforcement Learning: State-of-the-Art*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 45–73.
7. Kumar, A.; Zhou, A.; Tucker, G.; Levine, S. Conservative q-learning for offline reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1179–1191.
8. Nakamoto, M.; Zhai, Y.; Singh, A.; Mark, M.S.; Ma, Y.; Finn, C.; Kumar, A.; Levine, S. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *arXiv* **2023**, arXiv:2303.05479.
9. Ha, D.; Schmidhuber, J. World models. *arXiv* **2018**, arXiv:1803.10122.
10. Hafner, D.; Lillicrap, T.; Ba, J.; Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv* **2019**, arXiv:1912.01603.
11. Hafner, D.; Lillicrap, T.; Norouzi, M.; Ba, J. Mastering atari with discrete world models. *arXiv* **2020**, arXiv:2010.02193.
12. Wu, P.; Escontrela, A.; Hafner, D.; Abbeel, P.; Goldberg, K. Daydreamer: World models for physical robot learning. In Proceedings of the Conference on Robot Learning, PMLR, Atlanta, GA, USA, 6–9 November 2023; pp. 2226–2240.
13. Fujimoto, S.; Gu, S.S. A minimalist approach to offline reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 20132–20145.
14. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
15. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv* **2018**, arXiv:1801.01290.
16. Rajeswaran, A.; Kumar, V.; Gupta, A.; Vezzani, G.; Schulman, J.; Todorov, E.; Levine, S. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv* **2017**, arXiv:1709.10087.