Proceedings of Machine Learning Research vol 242:1107–1121, 2024 6th Annual Conference on Learning for Dynamics and Control

Residual Learning and Context Encoding for Adaptive Offline-to-Online Reinforcement Learning

Mohammadreza Nakhaei[1] MOHAMMADREZA.NAKHAEI@AALTO.FI
Aidan Scannell[1,2] AIDAN.SCANNELL@AALTO.FI
Joni Pajarinen[1] JONI.PAJARINEN@AALTO.FI
[1]*Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland*
[2]*Finnish Center for Artificial Intelligence, Finland*

Editors: A. Abate, M. Cannon, K. Margellos, A. Papachristodoulou

Abstract

Offline reinforcement learning (RL) allows learning sequential behavior from fixed datasets. Since offline datasets do not cover all possible situations, many methods collect additional data during online fine-tuning to improve performance. In general, these methods assume that the transition dynamics remain the same during both the offline and online phases of training. However, in many real-world applications, such as outdoor construction and navigation over rough terrain, it is common for the transition dynamics to vary between the offline and online phases. Moreover, the dynamics may vary during the online fine-tuning. To address this problem of changing dynamics from offline to online RL we propose a residual learning approach that infers dynamics changes to correct the outputs of the offline solution. At the online fine-tuning phase, we train a context encoder to learn a representation that is consistent inside the current online learning environment while being able to predict dynamic transitions. Experiments in D4RL MuJoCo environments, modified to support dynamics' changes upon environment resets, show that our approach can adapt to these dynamic changes and generalize to unseen perturbations in a sample-efficient way, whilst comparison methods cannot[1].

Keywords: Adaptive RL, Offline-to-Online RL, Context Encoding

1. Introduction

Offline reinforcement learning (RL) (Levine et al., 2020; Prudencio et al., 2023) has the potential to learn policies to accomplish complicated tasks from offline data without interacting with the environment. However, the environment in which these policies are deployed can in practice differ from the environment where the data was collected. Therefore, a fine-tuning phase that makes the policies adaptive to different modifications to the environment is necessary for real-world applications. An example is hydraulic systems (Egli and Hutter, 2022) where temperature influences the properties of the system and adaptation can be crucial.

Residual learning enables learning a residual agent that corrects the actions of a base policy. Prior research has used residual learning to combine conventional feedback controllers with RL agents (Johannink et al., 2019; Rana et al., 2020; Zhang et al., 2022b) for manipulation and navigation tasks, leading to improved sample efficiency. In this work, we use residual learning to adapt offline policies to environments with differing dynamics.

1. Code available at: https://github.com/MohammadrezaNakhaei/ReLCE

To enable the residual agent to adapt to the changes in the environment, we use a context encoder where a short history of previous transitions is used to infer the changes in the environment. We use the context encoder to learn a latent variable indicating the changes in the dynamics. Our representation learning relies upon minimizing the error over multi-step predictions as well as accurately predicting other transitions.

In this paper, the experimental focus is on MuJoCo (Todorov et al., 2012) locomotion tasks from the D4RL (Fu et al., 2020) benchmark. We extend the environments such that during online fine-tuning we can resample transition dynamics' parameters upon the environment reset at the start of each episode. The assumption is that at the beginning of each episode, a new set of dynamics parameters is sampled from an unknown distribution. Once the dynamics parameters are sampled, the dynamics remain constant for the duration of an episode.

We show that our approach can adapt to different changes in dynamics parameters whilst considering the base offline policy and a short history of transitions. Further to this, we show that our approach generalizes to unseen dynamics parameters. That is, it generalizes to out-of-distribution changes in the environment that were not present during training.

2. Problem Statement

In this paper, we consider offline-to-online RL. However, in contrast to previous approaches, we do not assume that the transition dynamics remain the same during the offline and online training phases. More specifically, we assume that during each episode of the online training phase, the transition dynamics are governed by one of N_M sets of transition dynamics parameters. The goal is to train an agent that can adapt to these dynamics changes using only a limited number of interactions.

We assume that the offline dataset is collected from a *Markov Decision Process* (MDP) $M_{\text{offline}} = \langle S, A, R, P_{\text{offline}}, \gamma, \rho_0 \rangle$ consisting of state space S, action space A, a scalar reward function R, transition dynamics $P_{\text{offline}}(s_{t+1}|s_t, a_t)$ which represent the distribution of possible next states conditioned on the current state and action, a discount factor $\gamma \in [0, 1]$ and an initial state distribution $\rho_0(s_0)$. A single trajectory (episode) consists of the list of states, actions, and rewards $\tau = [(s_1, a_1, r_1), (s_2, a_2, r_2), ..., (s_{T-1}, a_{T-1}, r_{T-1}), (s_T)]$ where s_T is the termination state. In online interactions, similar to previous meta-RL formulations, we assume a distribution of MDPs $p(M)$ with shared state space, action space, and reward function while the transition dynamics $P_i(s_{t+1} \mid s_t, a_t)$ vary between different MDPs. Each MDP is given by $M_i = \langle S, A, R, P_i, \gamma, \rho_0 \rangle$. At the beginning of each trajectory in the online fine-tuning stage, an MDP M_i is sampled from the distribution $p(M)$ and is consistent until the next trajectory. The objective is to find the policy π that maximizes the expected cumulative reward

$$J(\pi) = \mathbb{E}_{M_i \sim p(M), s_0 \sim \rho_0(s_0), s_{t+1} \sim P_i(s_t, a_t), a_t \sim \pi} \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right] . \tag{1}$$

3. Related Work

In this paper, we propose a novel challenge at the intersection of offline-to-online RL and adaptive RL. In this section, we first present an overview of offline RL methods since we use an offline policy as the base policy in the residual learning framework. Then we discuss different approaches for offline-to-online RL and compare them to our setting. Finally, we describe previous research on adaptive RL and illustrate how our approach is different and unique.

3.1. Offline Reinforcement Learning

Offline RL algorithms try to learn a policy that maximizes expected cumulative reward while only using static datasets without further interaction with the environment. The challenge in offline RL is distribution shift where the learned policy deviates from the behavior policy (the policy used to collect data) and selects out-of-distribution (OOD) actions for bootstrapping. To mitigate this, several methods constrain the policy to stay close to the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019; Wu et al., 2019; Siegel et al., 2019; Fujimoto and Gu, 2021). Another solution is to train pessimistic value functions (Kumar et al., 2020; Yu et al., 2020; Kidambi et al., 2020; Yu et al., 2021; An et al., 2021; Jeong et al., 2022; Lyu et al., 2022; Chen et al., 2023) where regularization is applied to penalize the action value functions for OOD actions. Another class of methods uses in-sample learning (Peng et al., 2019; Nair et al., 2020; Kostrikov et al., 2021; Hansen-Estruch et al., 2023; Xu et al., 2022) where only the actions in the datasets are considered for training value functions/policy and perform weighted imitation learning on the behavior policy. Other methods use conditional generative modelling (Chen et al., 2021; Janner et al., 2021; Zheng et al., 2022; Janner et al., 2022; Ajay et al., 2022) to learn policies from offline datasets, sidestepping the need for bootstrapping and learning value functions.

3.2. Offline-to-Online Reinforcement Learning

Pre-training on large datasets followed by fine-tuning on down-stream tasks has been investigated in modern machine learning, *e.g.*, computer vision (Wang et al., 2021; Cai et al., 2022; Li et al., 2023) and natural language processing (NLP) (Kenton and Toutanova, 2019; Li et al., 2021; Hu et al., 2023). To improve the performance of well-trained offline policies, Nair et al. (2020) imitated actions with high advantage estimates, Lee et al. (2022) modified the sampling method to incorporate near on-policy offline data, Zhao et al. (2022) carefully adjusted the behavior cloning regularization weight, and Zhao et al. (2023); Wen et al. (2023) used an ensemble of value functions whilst considering uncertainty and smoothness. Recently Zhang et al. (2022a) proposed using policy expansion sets where an offline policy is fixed and new policies are trained; adaptive composition of policies is used to interact with the environment. This work is close to our method since the offline policy is fixed during online fine-tuning. Still, all these methods assume that the environment during online adaptation is fixed and consistent with the offline dataset. In contrast, our method learns an adaptive policy during online fine-tuning. Hybrid RL (Niu et al., 2022) considers imperfect simulators with offline data from the real environment and uses both to learn a policy, but the learned policy is not adaptive.

3.3. Adaptive Reinforcement Learning

Adaptive RL aims at improving the generalization of policies across dynamic changes and different tasks. Yu et al. (2017) propose to learn an online system identification module using supervised learning and then train a universal policy considering the predicted system parameters. Kumar et al. (2021, 2022) instead, use a simulator while varying different parameters and learn an adaptation module by predicting the latent space of system parameters from a history of transitions. Guha and Annaswamy (2021); Cheng et al. (2022) incorporate adaptive control to estimate and compensate for changes in the dynamics, these methods require a nominal dynamic model and are restricted to the environment with Lagrangian mechanics without contacts. Meta-RL methods based on *Model*

Agnostic Meta Learning (MAML) (Finn et al., 2017) learn a pre-trained model from a set of environments and adapt to a new environment within several updates (Nagabandi et al., 2019). Methods based on memory use past interactions to update the policy. In *PEARL* (Rakelly et al., 2019), a context encoder is learned from previous transitions to facilitate learning the action value function. Lee et al. (2020) proposed *Context-aware Dynamic Model* (CaDM) to learn a latent vector from previous transitions and use the latent space in the dynamic model for more accurate predictions. Seo et al. (2020) incorporate multiple choice learning in learning context-aware dynamic model. In Evans et al. (2022), N random transitions (s, a, s') from a single environment (trajectory) are passed through the encoder to infer the context accordingly. These works are similar to our method in the regard that prediction error is used to train the context encoder, but our method considers multi-step prediction loss, and future/past prediction loss, and also uses similarity loss to encourage consistency of the latent space in the same environment. In addition, our framework considers an offline fixed policy as the base policy and trains the residual agent on top of that.

4. Method

Given an offline policy π_{offline}, we propose an adaptive policy that learns a residual policy π_{residual} to account for errors in the offline policy. Our method consists of an offline agent, context encoder/decoder, and residual agent,

$$a_t = \alpha\pi_{\text{offline}}(a_t^{\text{offline}} \mid s_t) + (1-\alpha)\pi_{\text{residual}}(a_t^{\text{residual}} \mid s_t, a_t^{\text{offline}}, z_t) \tag{2}$$

$$z_t = e_\theta(s_{t-H:t-1}, a_{t-H:t-1}) \tag{3}$$

$$\hat{s}_{t+1} = d_\theta(s_t, a_t, z_t), \tag{4}$$

where $z_t \in R^d$ is a context variable, *i.e.* a latent variable indicating which MDP we believe we are in, α is a mixing coefficient, e_θ is the context encoder summarizing previous transitions, and d_θ is the decoder that predicts future states used for learning representations. In the remainder of this section, we discuss each aspect of our method in more detail. Fig. 1 provides an overview of our method and Alg. 1 summarizes the online fine-tuning procedure.

Offline Agent: We consider *Conservative Offline Model-Based policy Optimization* (COMBO) (Yu et al., 2021) for training the offline policy π_{offline}. This algorithm extends *Conservative Q-Learning* (CQL) (Kumar et al., 2020) by using samples from the learned ensemble of a probabilistic dynamic model to train a less conservative Q function. We used the same hyper-parameters as the original paper and trained the agent for one million gradient steps.

Context Encoder: We train the context encoder to infer the changes in the environment from a short history of previous transitions implicitly without knowing the dynamic parameters. To train the context encoder, we use the forward dynamic prediction error. The decoder takes the latent representation z_t, state s_t, and action a_t and predicts the next state s_{t+1}. To encourage learning a representation for long horizon predictions, we use a k step loss function, where we use the predicted state to make further predictions. We also consider predicting other transitions from the same trajectory. This enables the encoder to learn a representation that is useful for prediction along the same trajectory (same MDP). The transition is randomly selected from the same trajectory. Prediction objectives do not guarantee consistency of the latent space along a trajectory where the assumption is that during the online fine-tuning, dynamics changes occur along episodes. To address this issue, we add an objective to maximize the cosine similarity loss between latent vectors of the

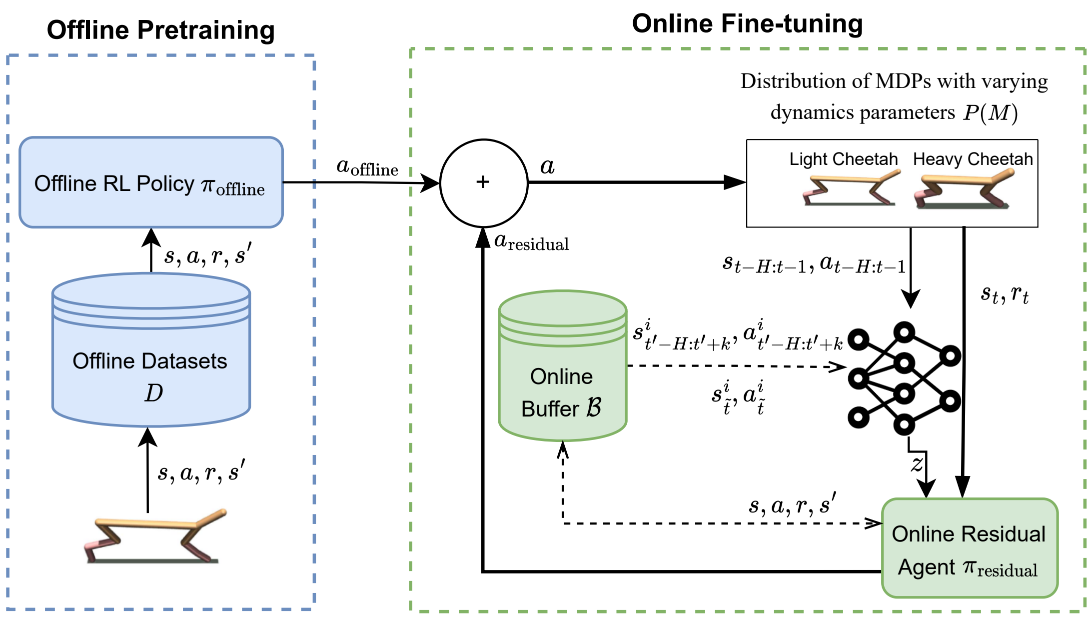


Figure 1: **RELCE overview** - The offline policy π_{offline} is used as base policy trained on existing datasets $\mathcal{D}$. The context encoder infers the changes in the environments, and the residual agent compensates for the modifications by considering the context and offline policy.

same environment. From the same trajectory in memory, we sample N sequence of transitions and compute latent vectors for each of them. We combine these objectives to obtain our encoder's objective:

$$\mathcal{L}(\{s^i_{t_i-H:t_i+k}, a^i_{t_i-H:t_i+k}, s^i_{t'_i}, a^i_{t'_i}, s^i_{t'_i+1}\}_{i=0}^N; \theta) = \underbrace{\sum_{i=1}^{N}\sum_{k=0}^{K-1} \gamma^k ||d_\theta(\hat{s}^i_{t_i+k}, a^i_{t_i+k}, z^i_{t_i}) - s^i_{t_i+k+1}||^2 +}_{k\text{-step predictions}}$$

$$\underbrace{\sum_{i=1}^{N} ||d_\theta(s^i_{t'_i}, a^i_{t'_i}, z^i_{t}) - s^i_{t'_i+1}||^2}_{\text{future/past prediction}} + \underbrace{\frac{\omega}{N-1} \sum_{i,j=1, i\neq j}^{N} -\frac{{z^i_{t_i}}^T}{||z^i_{t_i}||_2}\frac{z^j_{t_j}}{||z^j_{t_j}||_2}}_{\text{consistency}}, \quad (5)$$

where d_θ is the decoder which is trained along with the encoder, $\hat{s}$ represents the predicted states, ω is a hyper-parameter to balance consistency and prediction objectives. In the first step, we use the actual state to make the predictions *i.e.* $\hat{s}^i_t = s^i_t$.

Residual Agent: The residual agent aims to learn an adaptive policy that maximizes the expected cumulative reward across the distribution of MDPs by online interaction with the environments. At the beginning of each episode, we modify the dynamics by sampling from the distribution of MDPs. The transition dynamics then remain fixed until episode termination. We use a context encoder to infer the environment from a short history of previous transitions according to Eq. (3).

Algorithm 1 Adaptive online fine-tuning with residual agent

Require: trained offline agent π_{offline}, distribution of MDPs $p(M)$, context length H
Initialize context encoder e_θ, decoder d_θ, residual agent π_{residual} and replay buffer $\mathcal{B}$
for *step in training steps* **do**
  Sample an MDP from the distribution $M_i \sim p(M)$
  Set $t \leftarrow 0$ and observe the initial state s_0
  Initialize $S_{\text{episode}} = \{s_0\}$, $A_{\text{episode}} = \{\}$, $R_{\text{episode}} = \{\}$
  while *not done* **do**
    if $t < H$ **then**
      $a_t \sim \pi_{\text{offline}}(a_t|s_t)$
    else
      Compute context z_t according to Eq. (3)
      Compute the state of the residual agent s_t^{residual} according to Eq. (6)
      Get the total action a_t according to Eq. (2)
    end
    Interact with the environment with action a_t, observe the new state s'_t and the reward r_t
    Add new state s'_t, action a_t, and reward r_t to S_{episode}, A_{episode}, R_{episode} and set $t \leftarrow t+1$
    Sample training batch $\{s^i_{t'-H:t'}, a^i_{t'-H:t'}, s^i_{\tilde{t}}, a^i_{\tilde{t}}, r^i_{\tilde{t}}\}$ from the buffer $\mathcal{B}$
    Train context encoder using objective in Eq. (5) and train the residual agent using *SAC*
  end
  Add trajectory $(S_{\text{episode}}, A_{\text{episode}}, R_{\text{episode}})$ to the buffer $\mathcal{B}$
end

The residual agent observes the state of the environment, context vector, and the action of the offline policy,

$$s_t^{\text{residual}} = [s_t, a_t^{\text{offline}}, z_t]^T . \tag{6}$$

We train the residual agent to compensate for the offline policy and output corrective actions. In Eq. (2) α is a hyper-parameter with default value $\alpha = 0.75$ that chooses the importance of the offline policy vs. the residual policy.

We use the offline policy until the time step is more than the sequence length of the context encoder, *i.e.* $t = H$. Then we compute the context vector according to the prior transitions and determine the state of the residual agent according to Eq. (6). For training the residual agent, we use the *Soft Actor Critic* (SAC) algorithm (Haarnoja et al., 2018). The context encoder is fixed during the optimization of the Q-functions and the policy.

5. Experiments

We evaluate our approach on continuous control tasks with different datasets from the D4RL (Fu et al., 2020) benchmark. At the beginning of each episode, the mass of each link and the damping ratio of each joint are scaled by random numbers sampled from $[0.75, 0.85, 1, 1.15, 1.25]$ uniformly. We aim to answer the following questions:

- Can our context-aware residual agent learn to adapt to transition dynamics changes?
- Can our context encoder learn representations that enable the agent to predict future states across different dynamics parameters?

- Can our approach generalize to transition dynamics not seen during training?

5.1. Evaluation of the Adaptation Performance

In this section, we try to answer the first question and compare our methods adaptation performance to the baselines. The aim is to learn an adaptive policy during online fine-tuning with a limited number of interactions. We consider the following baselines:

- **Recurrent SAC** (Yang and Nguyen, 2021) uses recurrent networks for policy and value functions. We consider two variations: in the first variation we train the agent from scratch only by interacting online. In the second variation, we use residual learning with an offline policy similar to our method. We consider this baseline since it directly uses a history of transition to learn the policy and value function. In contrast, our method infers the dynamics' context z and trains an adaptive policy conditioned on this context.
- **Meta RL** algorithms learn an adaptive policy from a set of environments. **PEARL** (Rakelly et al., 2019) is an off-policy meta RL algorithm that includes a probabilistic context encoder. The comparison to this baseline can demonstrate the effect of offline policy and decoupling training the context encoder and policy learning.
- To demonstrate the necessity of adaptive online fine-tuning, we compare to **offline-to-online** methods. We consider **PEX** (Zhang et al., 2022a) and **Adaptive BC** (Zhao et al., 2022) for comparison.

For all the baselines, we use official implementations with our distribution of MDPs. We use one-dimensional *Convolution Neural Network* (CNN) to capture the temporal correlation between samples with $[4, 2, 1]$ kernel size followed by *ReLU* activation function. Convolution layers are followed by a linear layer that outputs the latent vector. For the decoder, we use *Multi-Layer preceptron* (MLP) networks with $[256, 256]$ hidden layers and *ReLU* activations. We also use Adam optimizer to train the encoder and decoder with a learning rate of 0.0001 while normalizing the target predictions. We use a default sequence length of $H = 10$, latent dimension of 8, and $K = 5$ step prediction loss for training the context encoder. We use $N = 4$ trajectories from the same environment and set ω to 0.1 to balance consistency and prediction objectives. We use MLP networks with $[256, 256]$ hidden layers followed by *ReLU* activations for the actors and critics networks of the residual agent. We use the Adam optimizer with learning rates of 0.0001 and 0.0003 to train actor and critic networks respectively.

We summarize the results in Table 1. Directly using recurrent networks in *SAC* without considering the offline agent has the worst performance and is not sample-efficient. Residual learning with the offline policy as the base improves the performance and sample-efficiency of *SAC* with recurrent networks. This suggests that a residual framework using the offline policy simplifies the problem and increases sample efficiency. *Residual RNN SAC* has a competitive performance in the *hopper* environment for different types of datasets, however, for other environments, it cannot learn the task within 250k time-steps. *PEARL* is more sample-efficient than *RNN SAC* and even has a better performance than *Residual RNN SAC* in *halfcheetah*, but within the sample budget, it cannot learn to adapt to different changes in the dynamics effectively. In the online fine-tuning phase, *PEX* and *Adaptive BC* improve performance when interacting with modified environments and learn more robust policies compared to the offline agent. Surprisingly, *Adaptive BC* outperforms our method

TASK	RNN SAC	RESIDUAL RNN SAC	PEARL	PEX	ADAPTIVE BC	RELCE (Ours)
hopper-medium-v2	35.64±10.04	65.24±12.03	43.37±8.03	69.98±28.83	102.55±0.99	94.74±0.88
hopper-medium-replay-v2	35.64±10.04	86.13±9.72	43.37±8.03	85.18±21.86	107.8±2.26	97.70±0.70
hopper-expert-v2	35.64±10.04	97.46±5.36	43.37±8.03	89.39±23.97	108.09±1.80	97.90±0.64
halfcheetah-medium-v2	9.22±0.69	40.58±4.28	64.65±8.03	62.98±3.08	83.40±3.35	92.86±2.50
halfcheetah-medium-replay-v2	9.22±0.69	43.14±1.73	64.65±8.03	53.67±1.70	78.88±3.19	93.93±3.47
halfcheetah-expert-v2	9.22±0.69	59.15±8.84	64.65±8.03	91.6±2.43	85.42±3.41	95.87±3.22
walker2d-medium-v2	13.00±2.64	32.83±8.60	16.63±7.11	80.60±16.53	72.96±8.60	90.80±3.66
walker2d-medium-replay-v2	13.00±2.64	51.76±9.46	16.63±7.11	87.50±9.98	96.15±4.08	92.82±2.04
walker2d-expert-v2	13.00±2.64	47.97±5.49	16.63±7.11	96.15±9.78	103.86±3.26	104.37±3.04
invertedpendulum-replay	92.82±7.11	98.17±3.32	74.73±16.85	76.40±7.20	72.67±9.57	100±0

Table 1: Results for adaptive online fine-tuning after 250k time-steps averaged over 10 random seeds, scores are normalized according to D4RL.

without considering any context or history in the *hopper* environment. We speculate that with different dynamic perturbations (changes in the mass and damping ratio) in the environment, the agent learns to behave conservatively and applies more torque/force trying to jump and move forward for different masses. To evaluate our speculation, we consider *invertedpendulum* environment and we scale the mass from $[0.25, 0.5, 0.75, 1, 1.5, 2, 3, 5]$ uniformly. We collect the dataset for the offline agent by training *SAC* for 200k timesteps and we use the replay buffer for the dataset. This environment is sensitive to the mass and simply applying more force/torque in different masses is not a solution. Methods that consider history including ours outperform offline-to-online methods demonstrating the necessity for adaptation.

Fig. 2 shows the learning curves for different methods for the *hopper* environment with different datasets. In the methods that use residual learning, there is a drop in performance at the initial stage of fine-tuning since the residual agent is initialized and still not trained.

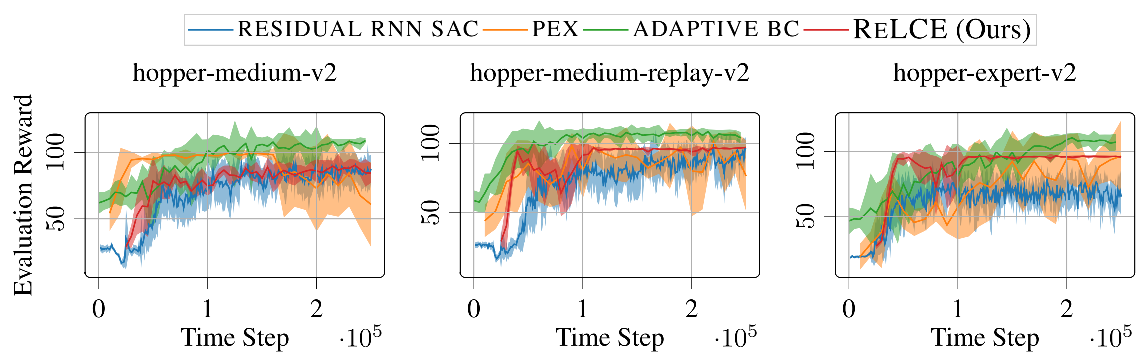
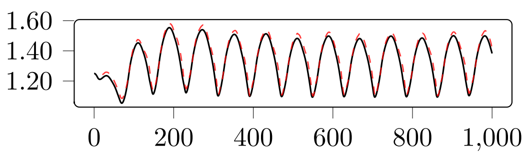


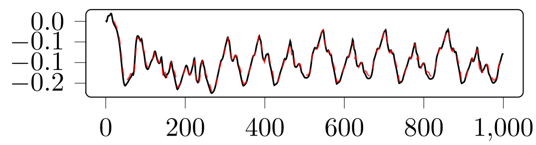
Figure 2: **Learning curves in hopper environment** Our method (red) is most sample efficient (*i.e.* converges in few environment steps) when the offline policy is learned with the expert data set (right) but still performs well with the medium data set (left). The shaded regions represent the standard deviation over 10 random seeds.

5.2. Latent Space Evaluation

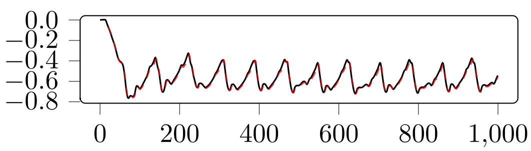
To evaluate the context encoder, we consider 10-step state predictions according to the latent space and the decoder. Fig. 3 shows predictions along a single trajectory for a randomly modified *hopper* environment. The 10-step future predictions for different states of the environment are close to the observed state even though we use 5 future steps for training. This indicates that the learned representation by the encoder can infer the changes in the environment.



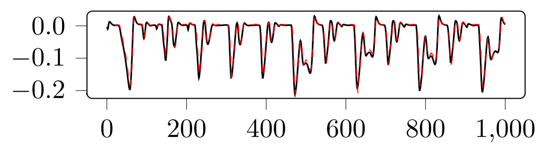
(a) First state dimension s_1



(b) Second state dimension s_2

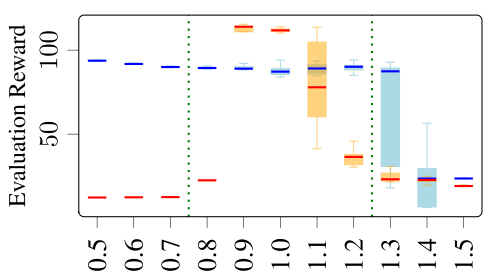


(c) Third state dimension s_3

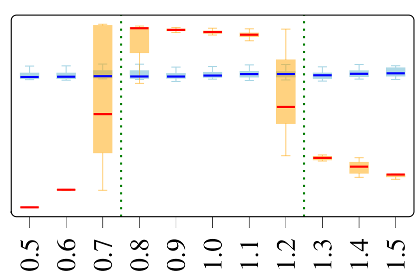


(d) Fourth state dimension s_4

Figure 3: **Multi-step predictions** RELCE makes accurate state predictions (10 steps) using the context encoder (red dashed), when compared to the ground truth (black).

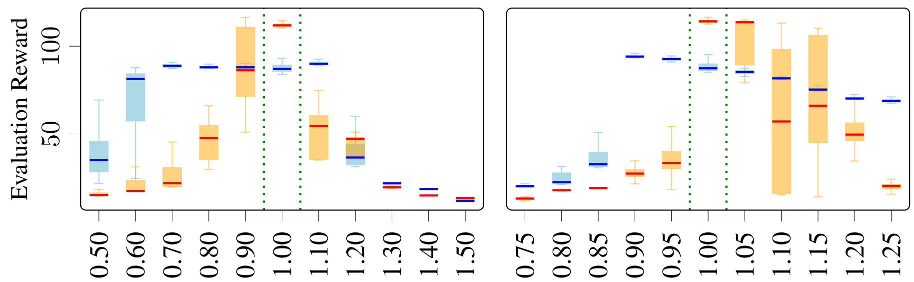


(a) Mass scale



(b) Damping scale

Figure 4: **Performance in the presence of dynamics changes** - Our method (blue) maintains good performance over a wider range of dynamics parameters than *Adaptive BC* (red) on the hopper-medium-replay task. Green lines separate in-distribution and out-of-distribution changes. Boxes represent the interquartile range (IQR) with median.



(a) Friction scale (b) Foot length scale

Figure 5: **Generalization to out-of-distribution dynamics parameters** - Our method (blue) shows some generalization to dynamics parameters outside of the distribution used during the online training on hopper-medium-replay. In contrast, *Adaptive BC* (red) struggles to generalize outside the training distribution. Green lines separate in-distribution and out-of-distribution changes. Boxes represent the interquartile range (IQR) with median.

5.3. Generalization to Unseen Dynamics

In this section, we investigate if our approach can adapt to changes in the environment that were not included during online fine-tuning. First, we investigate the same type of changes but with different magnitudes. Fig. 4 shows how different mass and damping ratios affect our method's performance, in terms of evaluation reward. We evaluate our method and *Adaptive BC* 100 times for each change in the environment. While *Adaptive BC* has a better performance for changes included in the training, especially small changes, our method can generalize to out-of-distribution changes. We speculate that for higher values of mass, the limitation in the actions (torques) makes it impossible for the agent to perform the task.

Next, we consider different dynamics changes that were not used in training. To this end, we change the friction coefficients for the joints and the foot length. Fig. 5 represent the results for our method and *Adaptive BC*. Our method outperforms *Adaptive BC* on almost all of the modified environments and is more stable with less variance in performance. This demonstrates that our method can adapt to different changes, even if the changes did not happen at training.

6. Conclusion

In this paper, we propose the novel problem of adaptive offline-to-online RL where the dynamics can change at each episode during online fine-tuning. We present a residual learning framework with context encoding to train an adaptive policy. In contrast to previous offline-to-online RL approaches, our method can compensate for dynamics changes by considering a short history of state transitions from the environment. Moreover, our experiments demonstrated that it can generalize to out-of-distribution dynamics that were not present in the online fine-tuning stage.

Future work We believe that our method can be improved in multiple ways. For instance, we use a constant coefficient for balancing the importance of the offline policy and the adaptive residual policy. However, automatically tuning this hyper-parameter could alleviate the drop in performance at the early stages of training.

Acknowledgments

We acknowledge CSC – IT Center for Science, Finland, for awarding this project access to the LUMI supercomputer, owned by the EuroHPC Joint Undertaking, hosted by CSC (Finland) and the LUMI consortium through CSC. We acknowledge the computational resources provided by the Aalto Science-IT project. J. Pajarinen was partly supported by Research Council of Finland (345521). M. Nakhaei was supported by Business Finland (BIOND4.0 - Data Driven Control for Bioprocesses). A. Scannell was supported by the Research Council of Finland from the Flagship program: Finnish Center for Artificial Intelligence (FCAI).

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B Tenenbaum, Tommi S Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2022.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Likun Cai, Zhi Zhang, Yi Zhu, Li Zhang, Mu Li, and Xiangyang Xue. Bigdetection: A large-scale benchmark for improved object detector pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4777–4787, 2022.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Liting Chen, Jie Yan, Zhengdao Shao, Lu Wang, Qingwei Lin, and Dongmei Zhang. Conservative state value estimation for offline reinforcement learning. *arXiv preprint arXiv:2302.06884*, 2023.

Yikun Cheng, Pan Zhao, Fanxin Wang, Daniel J. Block, and Naira Hovakimyan. Improving the robustness of reinforcement learning policies with $\updownarrow_1$ adaptive control. *IEEE Robotics and Automation Letters*, 7(3):6574–6581, 2022. doi: 10.1109/LRA.2022.3169309.

Pascal Egli and Marco Hutter. A general approach for the automation of hydraulic excavator arms using reinforcement learning. *IEEE Robotics and Automation Letters*, 7(2):5679–5686, 2022.

Ben Evans, Abitha Thankaraj, and Lerrel Pinto. Context is everything: Implicit identification for dynamics adaptation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2642–2648. IEEE, 2022.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

Anubhav Guha and Anuradha M. Annaswamy. Online policies for real-time control using mrac-rl. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 1808–1813, 2021. doi: 10.1109/CDC45484.2021.9683641.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

Jihwan Jeong, Xiaoyu Wang, Michael Gimelfarb, Hyunwoo Kim, Scott Sanner, et al. Conservative bayesian model-based value expansion for offline policy optimization. In *The Eleventh International Conference on Learning Representations*, 2022.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6023–6029. IEEE, 2019.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2021.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Ashish Kumar, Zhongyu Li, Jun Zeng, Deepak Pathak, Koushil Sreenath, and Jitendra Malik. Adapting rapid motor adaptation for bipedal robots. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1161–1168. IEEE, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 5757–5766. PMLR, 2020.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR, 2022.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Ming Li, Jie Wu, Xionghui Wang, Chen Chen, Jie Qin, Xuefeng Xiao, Rui Wang, Min Zheng, and Xin Pan. Aligndet: Aligning pre-training and fine-tuning in object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6866–6876, 2023.

Shiyang Li, Semih Yavuz, Wenhu Chen, and Xifeng Yan. Task-adaptive pre-training and self-training are complementary for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1006–1015, 2021.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Haoyi Niu, Yiwen Qiu, Ming Li, Guyue Zhou, Jianming HU, Xianyuan Zhan, et al. When to trust your simulator: Dynamics-aware hybrid offline-and-online reinforcement learning. *Advances in Neural Information Processing Systems*, 35:36599–36612, 2022.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Rafael Figueiredo Prudencio, Marcos R. O. A. Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–0, 2023. doi: 10.1109/TNNLS.2023.3250269.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

Krishan Rana, Ben Talbot, Vibhavari Dasagi, Michael Milford, and Niko Sünderhauf. Residual reactive navigation: Combining classical and learned navigation strategies for deployment in unknown environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11493–11499. IEEE, 2020.

Younggyo Seo, Kimin Lee, Ignasi Clavera Gilaberte, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel. Trajectory-wise multiple choice learning for dynamics generalization in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:12968–12979, 2020.

Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavior modelling priors for offline reinforcement learning. In *International Conference on Learning Representations*, 2019.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2021.

Xiaoyu Wen, Xudong Yu, Rui Yang, Chenjia Bai, and Zhen Wang. Towards robust offline-to-online reinforcement learning via uncertainty and smoothness. *arXiv preprint arXiv:2309.16973*, 2023.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. In *The Eleventh International Conference on Learning Representations*, 2022.

Zhihan Yang and Hai Nguyen. Recurrent off-policy baselines for memory-based continuous control. *Deep RL Workshop, NeurIPS 2021*, 2021.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022a.

Ruiqi Zhang, Jing Hou, Guang Chen, Zhijun Li, Jianxiao Chen, and Alois Knoll. Residual policy learning facilitates efficient model-free autonomous racing. *IEEE Robotics and Automation Letters*, 7(4):11625–11632, 2022b.

Kai Zhao, Yi Ma, Jinyi Liu, HAO Jianye, Yan Zheng, and Zhaopeng Meng. Improving offline-to-online reinforcement learning with q-ensembles. In *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

Yi Zhao, Rinu Boney, Alexander Ilin, Juho Kannala, and Joni Pajarinen. Adaptive behavior cloning regularization for stable offline-to-online reinforcement learning. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2022.

Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pages 27042–27059. PMLR, 2022.