Pre-training of Deep RL Agents for Improved Learning under Domain Randomization

Reinforcement Learning 2024-2025

Vincenzo Crisà 2143080

SAPIENZA
UNIVERSITÀ DI ROMA

1. Introduction

- Learning from pixels
- Domain Randomization
- DeepMind control suite

SAPIENZA
UNIVERSITÀ DI ROMA

Learning from pixels

1 Introduction

Common practice:

- MDP problem by stacking several consecutive images into a state;
- Encoder learns a latent representation of the images;
- Latent vector as input for a RL policy network.

This approach is known to have issues related to

- sample efficiency,
- degenerated feature representations,
- overfitting.

Image augmentation techniques improves performance by enhancing robustness and reducing overfitting without requiring any change to the RL algorithm.

Domain Randomization

1 Introduction

Bridge the visual gap between simulated training environmet and real world applications.

In visual domain randomization, the rendered observations from simulated environments are subjected to:

- Random texturing
- Image augmentation

This helps to learn a policy invariant to these shifts and is more likely to succeed on the transferring phase.

DeepMind control Suite

Google DeepMind's software stack for physics-based simulation and Reinforcement Learning environments, using MuJoCo physics.

The environments are characterized by

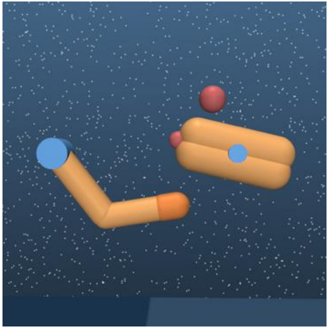
- *Domain* with a specific *Task*
- Observations are rgb 84x84 images
- Continuous action space [-1,1]
- Fixed episode length of 1000 steps
- Reward for episode [0,1000]

To provide the agent with a temporal perception of the environment, the images are stacked into a sequence of length 3 using a Wrapper.

DeepMind control Suite

1 Introduction

Domain: Finger, Task: Spin



Domain: Walker, Task: Walk

2. Data-regularized Q (DrQ)

- DrQ
- Action decision process
- Training parameters
- Training statistics
- Training results

SAPIENZA
UNIVERSITÀ DI ROMA

DrQ

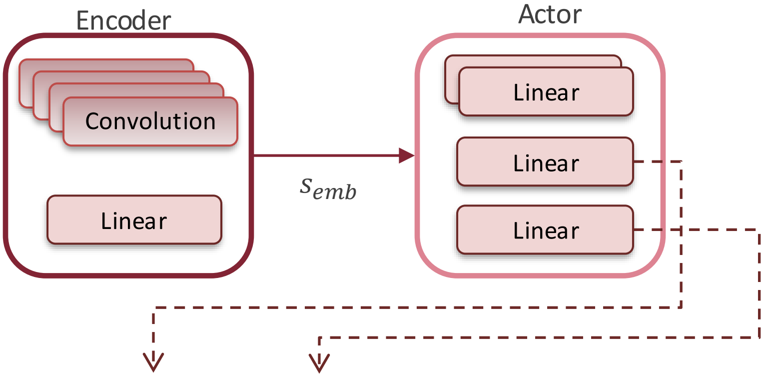
2 Data-regularized Q

DrQ enhance the Soft Actor-Critic training by introducing separate regularization mechanisms:

- Image augmentation on the images sampled from the replay buffer:
  - Padding each size by 4 pixels
  - Random crop back to the original resolution, shifted by ± 4 pixels
- Averaging the Q function for two different augmentation of the same state.

Action decision process

2 Data-regularized Q

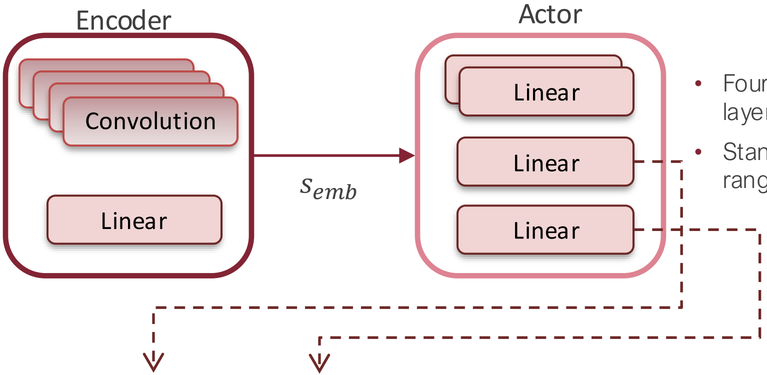


$$a_t = \tanh(\mu_\theta(s_{emb}) + \sigma_\theta(s_{emb})\epsilon) \text{ where } \epsilon \sim \mathcal{N}(0,1)$$

Action decision process

2 Data-regularized Q

- Four convolution layers with 3x3 kernel size and 32 channels
- Linear layer with output size set to 50



- Four fully connected layers of size 1024
- Standard deviation range: [-10,2]

$$a_t = \tanh(\mu_\theta(s_{emb}) + \sigma_\theta(s_{emb})\epsilon) \text{ where } \epsilon \sim \mathcal{N}(0,1)$$

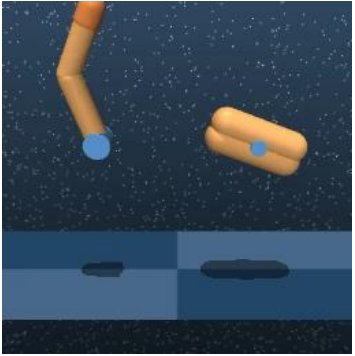
Training parameters

2 Data-regularized Q

Parameter	Value
Environment steps	300,000
Learning rate	10^{-3}
Batch size	128
Optimizer	Adam
Initial temperature	0.1
Soft target update rate	0.01
Discount factor	0.99
Device	mps

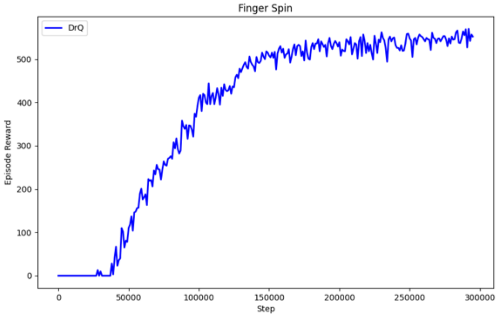
Training statistics

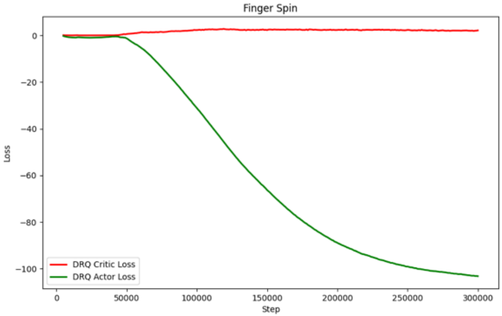
2 Data-regularized Q



Training statistics

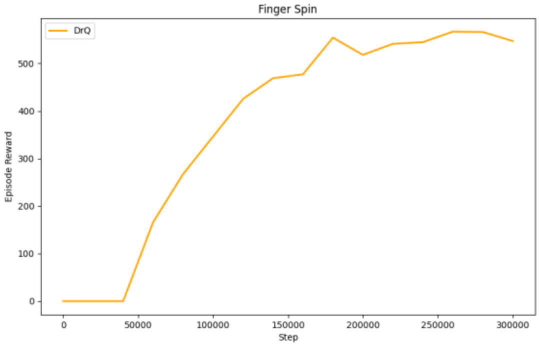
2 Data-regularized Q

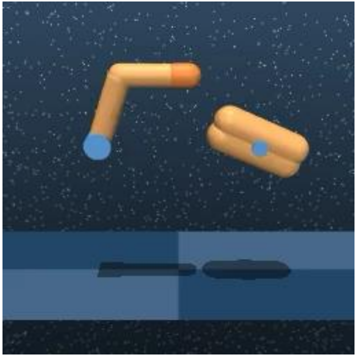




Training results

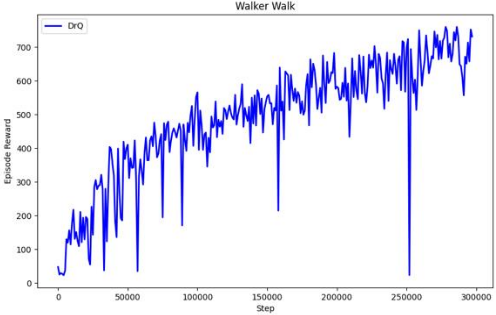
2 Data-regularized Q

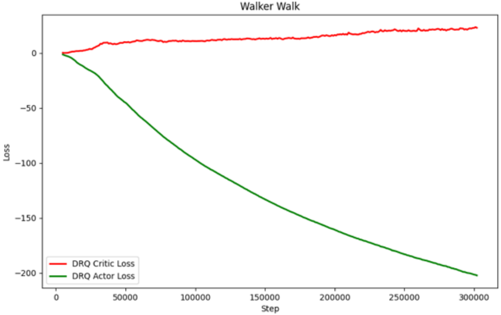




Training statistics

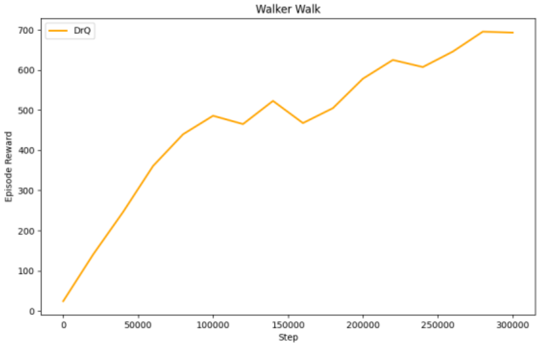
2 Data-regularized Q

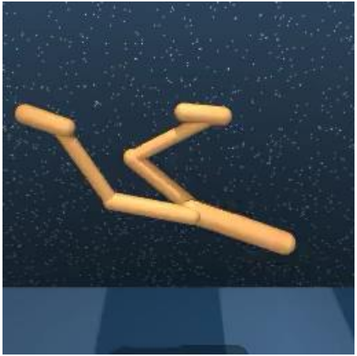




Training results

2 Data-regularized Q





3. Domain Randomization Adjusting Pre-training (DRAP)

- DRAP
- Domain Randomization Removal architecture
- Sequence dataset creation
- Pre-training parameters

DRAP

3 Domain Randomization Adjusting Pre-training

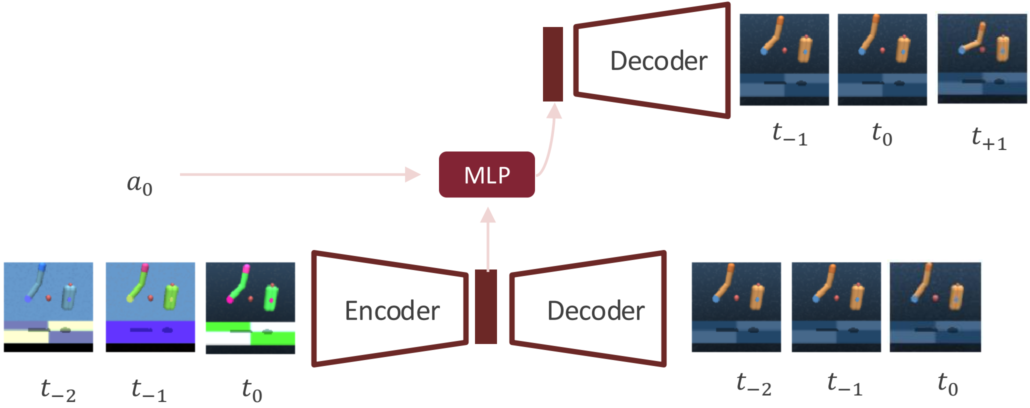
Domain Randomization Adjusting Pre-training (DRAP) adds additional variance from visual randomization separating visual pre-training to achieve visual invariance properties from RL training.

- Use the simulation to construct paired images of randomized observations and canonical observations.
- Train an Encoder-Decoder network to reconstruct the canonical observation and predict the future canonical observation from the randomized images.
- Use the pre-trained perception encoder as initialization for the encoder in DrQ.

The encoder learns to focus on the core parts of the images.

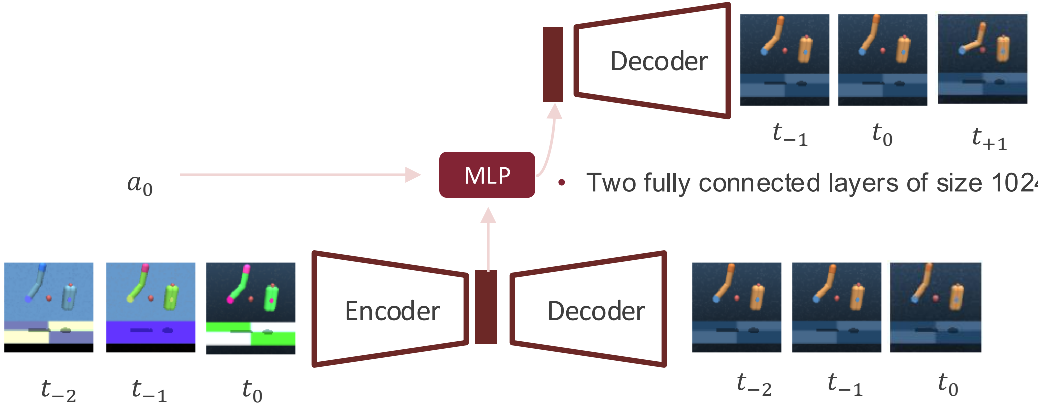
Domain Randomization Removal architecture

3 Domain Randomization Adjusting Pre-training



Domain Randomization Removal architecture

3 Domain Randomization Adjusting Pre-training



- Two fully connected layers of size 1024

- One upconvolution layer with 3x3 kernels and stride 2
- Two convolutions with 3x3 kernel

Sequence dataset creation

3 Domain Randomization Adjusting Pre-training

The datased is characterized by sequences of observation sampled using the mujoco framework from the DeepMind control suite.

It allows to render the observation directly from the base XML file.

Each source file is characterized by two common XML files that defines

- the sky,
- material textures.

To create a more diverse and robust dataset, starting from the canonical XML file, three randomized versions are generated by modifying common simulation characteristics.

Sequence dataset creation

3 Domain Randomization Adjusting Pre-training

```
<mujoco>
  <asset>
      <texture name="skybox" type="skybox" builtin="gradient" rgb1=".4 .6 .8"
rgb2="0 0 0"
               width="800" height="800" mark="random" markrgb="1 1 1"/>
  </asset>
</mujoco>
```

builtin field:

- "*gradient*" with 70% of probability
- "*flat*" with 30% of probability

markrgb value is randomly generated

Sequence dataset creation

3 Domain Randomization Adjusting Pre-training

```xml
<mujoco>
  <asset>
    <texture name="grid" type="2d" builtin="checker" rgb1=".1 .2 .3" rgb2=".2 .3 .4" width="300"
height="300" mark="edge" markrgb=".2 .3 .4"/>
    <material name="grid" texture="grid" texrepeat="1 1" texuniform="true" reflectance=".2"/>
    <material name="self" rgba=".7 .5 .3 1"/>
    <material name="self_default" rgba=".7 .5 .3 1"/>
    <material name="self_highlight" rgba="0 .5 .3 1"/>
    <material name="effector" rgba=".7 .4 .2 1"/>
    <material name="effector_default" rgba=".7 .4 .2 1"/>
    <material name="effector_highlight" rgba="0 .5 .3 1"/>
    <material name="decoration" rgba=".3 .5 .7 1"/>
    <material name="eye" rgba="0 .2 1 1"/>
    <material name="target" rgba=".6 .3 .3 1"/>
    <material name="target_default" rgba=".6 .3 .3 1"/>
    <material name="target_highlight" rgba=".6 .3 .3 .4"/>
    <material name="site" rgba=".5 .5 .5 .3"/>
  </asset>
</mujoco>
```

Sequence dataset creation

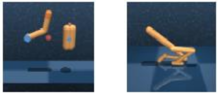
3 Domain Randomization Adjusting Pre-training

Each collected sequence is stored following the format:

- "*canonical*": images from the canonical XML, used as reference sequence

- "*randomized*": images randomly selected from the three randomized XMLs, used as input

- "*action*": the joints values used at t_0 to simulate the action
- "*future_canon*": the future canonical image corresponding to t_{+1}



Sequence dataset creation

To capture meaningful representations of the environment's behavior, data sequences are collected by simulating 1000 steps in canonical and randomized versions.

1. A random initial position and velocity are assigned to the agent.
2. To simulate an action, the agent's joints values in the XML file are randomized using values derived from $\mathrm{acos}(\phi)$ where ϕ is uniformly sampled from $[0, \pi]$ and a is a random integer value in [0,50].
3. Every 5 steps, a sequences of images is sampled to balance computational cost and data diversity.

To build a robust dataset, the randomized sequence elements are randomly selected from one of the three randomized environments, ensuring variability across the sampled data.

Pre-training Parameters

3 Domain Randomization Adjusting Pre-training

Parameter	Value
Environment steps	100,000
Learning rate	10^{-3}
Batch size	128
Optimizer	Adam

The images from the dataset are subjected to the same augmentation used in DrQ.

4. DrQ + DRAP

- Last part of the training
- Training statistics
- Training results
- Conclusions

SAPIENZA
UNIVERSITÀ DI ROMA

Last part of the training

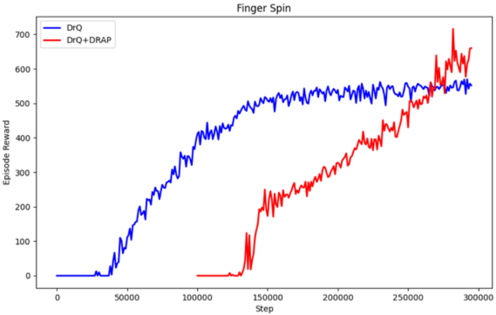
4 Drq+DRAP

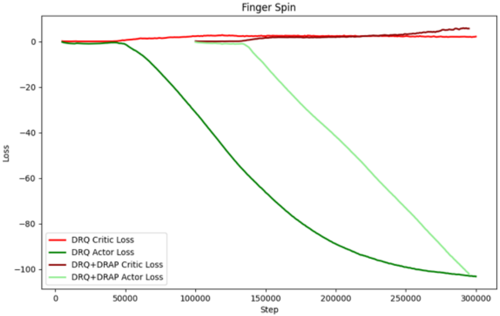
After the pre-training the decoder part is discarded, and the pre-trained encoder, is directly fine-tuned during DrQ training.

Since the previous training have been done on 300,000 environment steps, the pre-trained encoder's weights are used to initialize DrQ encoder and use it for the remaining 200,000 environment interactions.

Training statistics

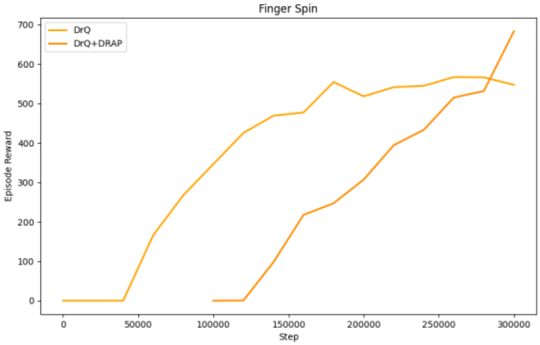
4 Drq+DRAP

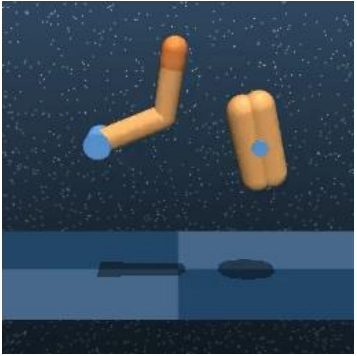




Training results

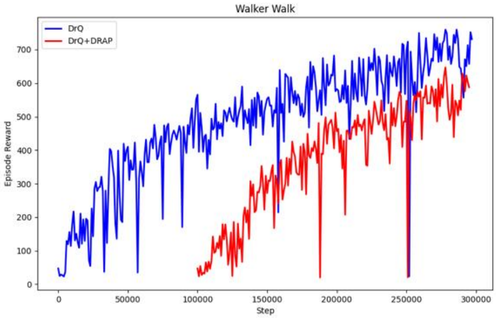
4 Drq+DRAP

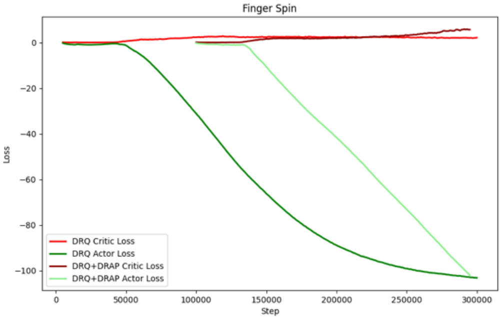




Training statistics

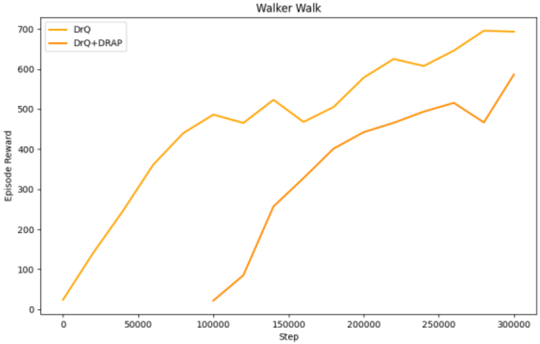
4 Drq+DRAP





Training statistics

4 Drq+DRAP



Training results

4 Drq+DRAP

Finger domain

	DrQ			DrQ + DRAP		
	Step	Reward	Duration	Step	Reward	Duration
Train	300,000	552	93.2 s	200,000	**660**	57.9 s
Evaluation	300,000	547		200,000	**683**	

Training results

4 Drq+DRAP

Walker domain

	DrQ			DrQ + DRAP		
	Step	Reward	Duration	Step	Reward	Duration
Train	300,000	**657**	77.9 s	200,000	588	61.1s
Evaluation	300,000	**693**		200,000	587	

Conclusions

4 Drq+DRAP

- The warm-start boosts the DrQ performace as the encoder has already learned to ignore visual variations, focusing on the agent and providing a much cleaner signal to the successive parts of the network.
- The finger domain is simpler than the walker domain because it is more “static” and involves only two joints. This simplicity makes it easier for the network to learn a good policy, leading to more significant performance gains and improvements compared to the walker domain, which is more dynamic and requires additional training.
- This results are obtained with random randomizations of the environments and so changing specific details of simulation, the results can be better or even worse.

Thank you for the attention.