Few-shot Transfer Learning for Deep Reinforcement Learning on Robotic Manipulation Tasks

Yuanzhi He[0009−0007−8424−2654], Christopher D. Wallbridge[0000−0001−9468−122X], Juan D. Hernández[0000−0002−9593−6789], and Gualtiero B. Colombo[0000−0001−5883−8517]

School of Computer Science and Informatics, Cardiff University, Cardiff, UK
{HeY65, WallbridgeC, HernandezVegaJ, ColomboG}@cardiff.ac.uk

Abstract. Robot manipulation with simulation has become a mainstream approach in the robotics field recently. It entails lower risk and cost compared to direct training a real robot. Various physics engines, such as MuJoCo, offer simulated environments tailored for robot manipulation tasks. As the robotics field rapidly grows, model complexity and training times increase exponentially to meet the demands of diverse tasks. Solving this is challenging as it requires complex models and long training times. Deep Reinforcement Learning (DRL) is the current best-performing way to solve robot manipulation problems. However, although certain algorithms utilized automated curriculum learning to tackle multi-task robot manipulation problems, the models were still too complex to be solved with one training from scratch with acceptable accuracy and reasonable training time. To address this, we introduce a novel few-shot Transfer Learning (TL) technique for DRL that applies both Forward Transfer (FT) and Reverse Transfer (RT). TL facilitates breaking down a complex problem into easier-to-solve sub-problems and transferring the acquired knowledge to more complex ones. Our TL method appears able to accelerate the training process for all the MuJoCo Fetch tasks, while even improving performance by 20% and accelerating 85% for the most complex FetchSlide environment.

Keywords: Transfer Learning · Deep Reinforcement Learning · Robot Manipulation · Physics Engine.

1 Introduction

In recent years, robotic experiments are often conducted in simulated environments rather than real ones due to cost and risk concerns, as well as the need for enhanced control and efficiency [1]. OpenAI Gym [2] is an open-source library which was created to simplify access to most of the latest Reinforcement Learning (RL) algorithms and soon became a significant platform for robotic simulated environments in particular with manipulation tasks (Fetch, Hand) [2, 3]. Among the various physics engines introduced to build simulated environments for robot manipulation tasks [1], MuJoCo presents high flexibility and

accuracy for simulating robot dynamics and contact forces [4]. However, many robot training processes are still extremely time-consuming, thus often resulting in under-performing and data-dependent outcomes. Within recent Machine Learning (ML) developments in robotics, including control systems and motion planning, significant improvements have been brought about by Deep Reinforcement Learning (DRL) models, which are widely used during the training process. This involves models that are large, complex and increasingly sophisticated, leading to high resource demands and long training processes [5].

State-of-the-art DRL implementations in robotic manipulation tasks obtain good results, yet certain tasks, including FetchSlide, have only been partially solved or require excessively long times to accomplish [3, 6]. To enable a more efficient and better-performing training process, we introduce a Few-shot Transfer Learning (TL) approach, utilizing a set of robotic MuJoCo tasks as testbeds. Instead of starting the learning process from scratch, TL was introduced to break down a complex problem into sub-tasks and transfer the knowledge acquired from solving simpler primary tasks to increasingly complex ones. The scope of this study is to investigate the effectiveness of TL in accelerating the training process while maintaining or improving performance across all the benchmarks considered.

2 Related Work

Transfer Learning (TL) is a Machine Learning (ML) technique that involves leveraging useful features gained from solving one task to improve performance on a related task. It relates to a number of different subcategories including Imitation Learning (IL), Multi-Task Learning (MT) and Goal-oriented Learning. Current research shows great potential for TL in the ML field [8, 9]. The TL approach can be categorised into three classes: Zero-shot, Few-shot and Sample-efficient. Zero-shot TL enables the development of an agent that can be directly applied to the target domain without requiring any training interaction [10]. The Sample-efficient TL is designed to accelerate the learning process of models through the rational use of data, thereby reducing data requirements and training costs [11]. Unlike Zero-shot and Sample-efficient TL, Few-shot TL enables only a few interactions to keep more useful features and adapts more effectively to the specific characteristics of the target task [12].

To the best of our knowledge, the application of IL to robotics is mostly limited to approaches implementing zero-shot and few-shot transfer [9, 13, 14]. However, although IL learns high-quality behavioural policies directly from expert demonstrations, thus making it particularly suitable for tasks requiring precise execution, it is still expert-dependent and time-consuming. In addition, because of the high complexity of these types of robotic problems, traditional RL solutions are not able to achieve satisfactory results.

More recently, Deep Reinforcement Learning (DRL) has been introduced to empower robots with a greater awareness and understanding of the environment. DRL appears successful in learning adaptive strategies from interactions

with the environment, which makes it applicable to various robotic tasks [15]. Current mainstream DRL algorithms have been divided into two types: on-policy and off-policy algorithms [16, 17]. The latest on-policy DRL algorithms include Proximal Policy Optimization (PPO) [18], enabling better convergence and robustness of robots, especially in policy optimization. However, due to the scale of most robotic models being quite large, off-policy DRL algorithms including Soft Actor-Critic [19] and Truncated Quantile Critics (TQC) [20] were created to tackle this problem. Hindsight Experience Replay (HER) [21] was also introduced to help the challenges related to inaccuracies and large action spaces by replacing the desired goal with the achieved goal and storing their experience replay in the replay buffer [3]. To ensure a fair result comparison and make RL algorithms easy to use, the Stable-Baselines3 (SB3) and RLlib libraries were designed and assembled with most of the mainstream DRL algorithms and simulated environments, including robotic environments [22, 23].

Finally, a set of simulated MuJoCo Fetch environments (Reach, Push, PickAndPlace and Stack) were implemented with alternative Goal-Oriented approaches [7, 24, 25]. These are often based on intrinsically motivated and curiosity-driven rewards while other proposed solutions applied forms of Multi-task Learning primarily focused on training a single model with multiple tasks with shared knowledge flow [26, 27].

Although all the methods and techniques mentioned in this section were able to produce some improvements on the state of the art, as the complexity of RL algorithms grows exponentially there is still a need for enhancing the performance of the current single-environment techniques.

3 Proposed Approach

Our proposed approach aims to solve simulated robotic manipulation tasks using TL (transferring the weights and biases of the first layer to the target tasks) as an alternative to DRL model training in isolation from scratch. The experiments considered are the set of MuJoCo manipulation Fetch environments provided by OpenAI Gym [2]. These include FetchReach-v2 (reaching a target point), FetchPush-v2 (pushing a block to a target point), FetchSlide-v2 (hitting a puck to reach a target point on a long and slippery table) and FetchPickAndPlace-v2 (picking an object from a source point and placing it to a target point). RL Baselines3 Zoo is a benchmark with metrics created by SB3 [22] showing that TQC [20] currently has the best performance on success rate and rewards metrics on all of the above environments [19]. This method accepts states, actions and goals as the input of the 'critic' network to calculate the inputs to the hidden 'actor' layers while the related goals and rewards are computed applying the HER technique [3, 21].

Fig. 1 shows an example of the TL process with Forward (Single and Double) and Reverse Transfer with the FetchReach environment selected as the primary task. Unlike the weights and biases that need to be initialized as random values when applying a DRL model in isolation, in TL a proportion of the pre-trained

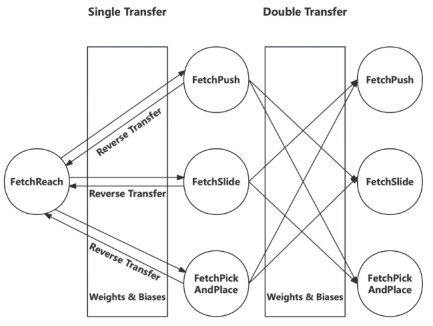

Fig. 1. Forward (Single, Double) and Reverse Transfer Learning.

parameters, i.e. weights and biases, are transferred between the models solving the primary and target tasks. To explain these operations in simpler terms we can observe that more complex tasks such as Pick-and-Place can be reduced to a combination of sub-tasks, one being the Reach tasks itself. Therefore if a model has already learnt through training how to complete the reaching action, then TL aims to exploit this already acquired knowledge so that the remaining training process (or indeed retraining) could only focus on the remaining sub-tasks. This is expected to require shorter training times and eventually also lead to gains in performance. It is worth noting that once some knowledge has been acquired, the related pre-trained parameters could be transferred to a number of different target tasks in parallel potentially producing further significant gains in runtime.

Existing research showing affinity to TL focuses either on trying to balance the training of the same model by using different tasks to compute the losses, similar to multi-objective optimization approaches (multi-task learning [26, 27]), or on exploiting the benefits of training a model through a sequence of increasingly complex tasks (goal-oriented computing [7, 24, 25]). On the contrary, the principal aim of our research switches the focus to the actual mechanisms used to transfer the knowledge, namely identifying what subsets of the pre-trained parameters need to be transferred, and how these stand in relation to the different tasks and sub-tasks. When only one transfer for each training is performed this is referred to as Single Transfer. Double Transfer obtains the useful features from a trained non-primary model to another non-primary model and finally Reverse Transfer transfers the useful features from a non-primary model back to the primary one. The latter seeks to validate to what extent the features learnt during more complex tasks will remain useful to solve basic ones (in other terms if solving complex tasks could cause some basic knowledge loss).

4 Experiments and Results

The MuJoCo Fetch environments have been implemented for TL. The latest version-2 release has been adopted because of its wider MuJoCo support. Because

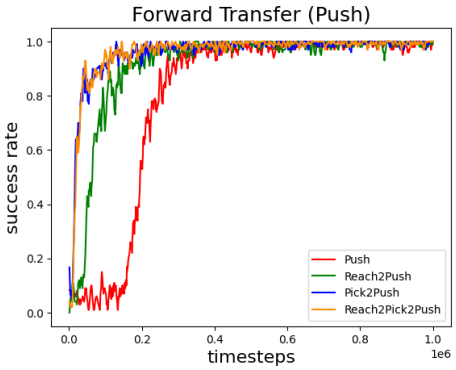

Fig. 2. Success rate vs timesteps for FetchPush-v2 with Single (Reach->Push and Pick->Push) and Double TL (Reach->Pick->Push) and without TL.

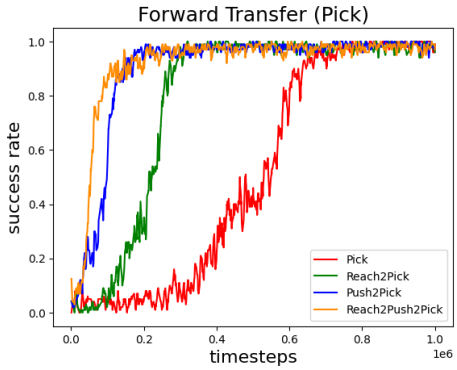

Fig. 3. Success rate vs timesteps for FetchPickAndPlace-v2 with Single (Reach->Pick and Push->Pick) and Double TL (Reach->Push->Pick) and without TL.

of its simplicity and reliability, the Stable-Baselines3 (SB3) has been utilized as a benchmark for performance comparison. To prevent uncertainty, all the experiments gathered the models with the best performance from 3 training sessions. The learning rate and batch size remain unchanged for all environments, which are 0.001 and 256 as suggested by the benchmark RL Baselines3 Zoo from SB3. We use TQC with HER as the main Deep Reinforcement Learning algorithm for environments with sparse rewards. The total training timesteps are different due to the complexity of each task. They are set to match the value of the benchmark SB3. The experiments were performed on a PC, with 64GB RAM, an Intel Core i7-14700K 2.5GHz CPU and an RTX 4060 GPU.

Forward Transfer Figures 2, 3 and 4 present the results of Single and Double Transfer from FetchReach to the other Fetch environments. By analyzing Figures 2 and 3, we can observe that the training approaches applying models pre-trained on both FetchPush and FetchPickAndPlace environments outperform those without any pre-training in runtime terms. Whether it is a Single or Double Transfer, from FetchPickAndPlace to FetchPush and vice versa, TL is always able to produce a faster convergence. Furthermore, by analyzing Fig. 4, on the FetchSlide task (the most complex of the Fetch environments), the model pre-trained with FetchPick is able to achieve the best overall performance by producing the highest success rate. It is worth noting that the pre-trained models with Double Transfer outperform those with Single Transfer, although even a simple pre-training with FetchReach produces a better performance than the model trained from scratch without any transfer applied.

Reverse Transfer Fig. 5 presents the result of Reverse Transfer from non-primary models (FetchPush, FetchSlide and FetchPickAndPlace) to FetchReach. It shows that the models pre-trained by FetchPush, FetchPickAndPlace and FetchSlide all outperform the original single-environment model by scoring success rates equal to or near to 1 compared with 0.91. However, the pre-trained

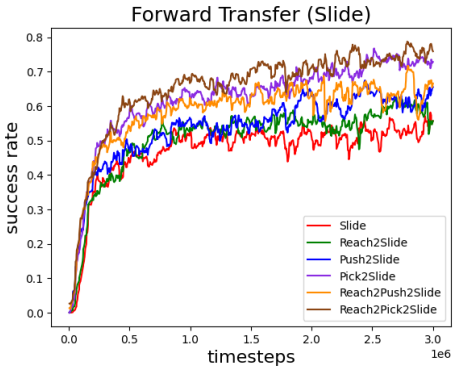

Fig. 4. Success rate vs timesteps for FetchSlide-v2 with Single (Reach->Slide, Push->Slide and Pick->Slide) and Double (Reach->Push->Slide and Reach->Pick->slide) TL and without TL.

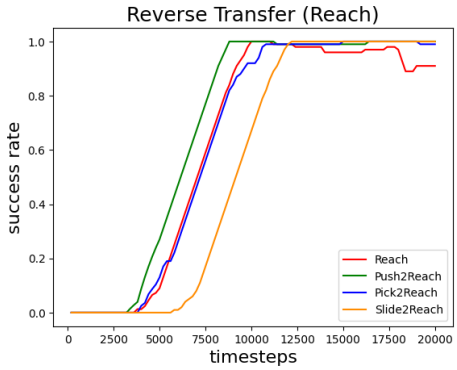

Fig. 5. Success rate vs timesteps for FetchReach-v2 with Reverse Transfer (Push->Reach, Pick->Reach and Slide->Reach) and without TL.

models from FetchPickAndPlace and FetchSlide converge slower than the model trained from scratch. This is because their complexity is higher than FetchPush indicating that a potential overtraining might have happened. As a result, models starting with pre-training on more complex tasks may eventually result in affecting the converging speed, although eventually leading to higher success rates.

5 Conclusion and Future Works

This study presents some preliminary results demonstrating encouraging improvements in applying both Forward (Single, Double) and Reverse TL to the robot training process. The proposed TL methods accelerate the original training process on single environments and are able to at least equalize the current benchmarks for each of the MuJoCo Fetch environments, while also scoring higher success rates for the most complex task (FetchSlide). This suggests great potential for further TL applications to simulated Robotic tasks. Our research aims to provide more insights on the actual mechanism of transfer, by focusing on which subsets of pre-trained parameters need to be transferred between tasks and how this can be achieved.

Future work will focus on the application of the TL method to more complex robotic environments (putting objects on a shelf [28]) and alternative physical engines such as Gazebo [29]. In addition, future research will also attempt to close the gap from simulation to real environments through the so-called Sim2Real Transfer [11]. Finally, we can also consider transfer between models solving the same task, albeit implementing different forms of rewards. These could be based on different principles, such as novelty and curiosity, or on auxiliary intermediate goals, for example generated using large language models [30, 31].

References

1. Erez, T., Tassa, Y. and Todorov, E.: Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In: 2015 IEEE international conference on robotics and automation (ICRA), pp. 4397–4404. IEEE (2015)
2. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: OpenAI Gym. In: arXiv preprint 2016. arXiv (2016)
3. Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., Kumar, V.: Multi-goal reinforcement learning: Challenging robotics environments and request for research. In: arXiv preprint 2018. arXiv (2018)
4. Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: IEEE/RSJ international conference on intelligent robots and systems 2012, pp. 5026–5033. IEEE, Vilamoura-Algarve (2012)
5. Soori, M., Arezoo, B. and Dastres, R.: Artificial intelligence, machine learning and deep learning in advanced robotics, a review. Cognitive Robotics **3**, 54–70 (2023)
6. Yang, X., Ji, Z., Wu, J., Lai, Y.K.: An open-source multi-goal reinforcement learning environment for robotic manipulation with pybullet. In: Annual Conference Towards Autonomous Robotic Systems 2021, Cham: Springer International Publishing, pp. 14–24. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-89177-0_2
7. Colas, C., Fournier, P., Chetouani, M., Sigaud, O., Oudeyer, P.Y.: Curious: intrinsically motivated modular multi-goal reinforcement learning. In: International conference on machine learning, pp. 1331–1340. PMLR (2019)
8. Pan, S.J., Qiang, Y.: A survey on transfer learning. In: IEEE Transactions on knowledge and data engineering 2009, vol. 22, pp. 1345–1359. IEEE (2009)
9. Jaquier, N., Welle, M.C., Gams, A., Yao, K., Fichera, B., Billard, A., Ude, A., Asfour, T., Kragić, D.: Transfer Learning in Robotics: An Upcoming Breakthrough? A Review of Promises and Challenges. In: arXiv preprint 2023. arXiv (2023)
10. Kirk, R., Zhang, A., Grefenstette, E., Rocktäschel, T.: A survey of zero-shot generalisation in deep reinforcement learning. Journal of Artificial Intelligence Research **76**, 201–264 (2023)
11. Zhu, Z., Lin, K., Jain, A.K., Zhou, J.: Transfer Learning in Deep Reinforcement Learning: A Survey. In: IEEE Transactions on Pattern Analysis And Machine Intelligence 2023, vol. 45, pp. 13344–13362. IEEE (2023)
12. Song, Y., Wang, T., Cai, P., Mondal, S.K., Sahoo, J.P.: A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. In: ACM Computing Surveys 2023, vol. 55, pp. 1–40. ACM (2023)
13. Hundt, A., Murali, A., Hubli, P., Liu, R., Gopalan, N., Gombolay, M., Hager, G.D.: " good robot! now watch this!": Repurposing reinforcement learning for task-to-task transfer. In: 5th Annual Conference on Robot Learning, pp. 1564–1574. PMLR (2022)
14. Jang, E., Irpan, A., Khansari, M., Kappler, D., Ebert, F., Lynch, C., Levine, S., Finn, C.: Bc-z: Zero-shot task generalization with robotic imitation learning. In: Conference on Robot Learning, pp. 991–1002. PMLR (2022)
15. Han, D., Mulyana, B., Stankovic, V., Cheng, S.: A survey on deep reinforcement learning algorithms for robotic manipulation. Sensors **23**(7), 3762 (2023)
16. Wang, H.N., Liu, N., Zhang, Y.Y., Feng, D.W., Huang, F., Li, D.S., Zhang, Y.M.: Deep reinforcement learning: a survey. Frontiers of Information Technology & Electronic Engineering **21**(12), 1726–1744 (2020)

17. Wang, X., Wang, S., Liang, X., Zhao, D., Huang, J., Xu, X., Dai, B., Miao, Q.: Deep reinforcement learning: A survey. In: IEEE Transactions on Neural Networks and Learning Systems. IEEE (2022)
18. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. In: arXiv preprint 2017. arXiv (2017)
19. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International conference on machine learning, pp. 1861–1870. PMLR (2018)
20. Kuznetsov, A., Shvechikov, P., Grishin, A., Vetrov, D.: Controlling overestimation bias with truncated mixture of continuous distributional quantile critics. In: International Conference on Machine Learning, pp. 5556–5566. PMLR (2020)
21. Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., Zaremba, W.: Hindsight experience replay. In: Advances in neural information processing systems on NeurIPS Proceedings. NIPS (2017)
22. Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N.: Stable-baselines3: Reliable reinforcement learning implementations. Journal of Machine Learning Research **22**(268), 1–8 (2021)
23. Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Goldberg, K., Gonzalez, J., Jordan, M. and Stoica, I.: RLlib: Abstractions for distributed reinforcement learning. In: International conference on machine learning, pp. 3053–3062. PMLR (2018)
24. Colas, C., Karch, T., Sigaud, O. and Oudeyer, P.Y.: Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. Journal of Artificial Intelligence Research **74**, 1159–1199 (2022)
25. Wang, R., Lehman, J., Clune, J., Stanley, K.O.: Poet: open-ended coevolution of environments and their optimized solutions. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 142–151. ACM (2019)
26. Zhang, Y. and Yang, Q.: A survey on multi-task learning. In: IEEE Transactions on Knowledge and Data Engineering, vol. 34, pp. 5586–5609. IEEE (2021)
27. Ruder, S.: An overview of multi-task learning in deep neural networks. arXiv preprint 2017. arXiv (2017)
28. Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K.O., Clune, J.: First return, then explore. Nature, **590**(7847), pp. 580–586 (2021)
29. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In 2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566), Vol. 3, pp. 2149–2154. IEEE (2004)
30. Du, Y., Watkins, O., Wang, Z., Colas, C., Darrell, T., Abbeel, P., Gupta, A., Andreas, J.: Guiding pretraining in reinforcement learning with large language models. In: International Conference on Machine Learning, pp. 8657–8677. PMLR (2023)
31. Colas, C., Teodorescu, L., Oudeyer, P.Y., Yuan, X. and Côté, M.A.: Augmenting autotelic agents with large language models. In: Conference on Lifelong Learning Agents, pp. 205–226. PMLR (2023)