Undergraduate Research Opportunity Program
(UROP) Project Report

Improving Multi-Step Model-Based Reinforcement Learning

By

New Jun Jie

Department of Computer Science

School of Computing

National University of Singapore

2020/2021

Undergraduate Research Opportunity Program
(UROP) Project Report

Improving Multi-Step Model-Based Reinforcement Learning

By

New Jun Jie

Department of Computer Science

School of Computing

National University of Singapore

2020/2021

Project No: U226020
Advisor: Dr. Harold Soh
Deliverables:
UROP CA Report

Table of Contents

Chapter 1

Objectives

"Reinforcement learning is learning what to do–how to map situations to actions–so as to maximize a numerical reward signal" [25]. Combined with high-capacity neural networks, reinforcement learning has shown promise in a range of highly complex sequential decision making problems, such as game playing and robotics. However, the sample complexity of most deep reinforcement learning algorithms is high, limiting its application in important domains.

Model-based reinforcement learning utilises an explicit model of environment dynamics to reduce the need for real-world data samples. In recent years, many model-based methods have been introduced that show promise in the range of things model-based reinforcement learning algorithms can do [8] [16]. However, recent benchmarking research show that a performance bottleneck of model-based methods below their model-free counterparts, also known as the dynamics bottleneck, and thus remains an open research problem. Therefore, model-based reinforcement learning is an important area of research in machine learning.

The objective of this research report is to detail the literature review conducted around improving multi-step model-based reinforcement learning, namely in consideration of the dynamics bottleneck, the research experiments conducted in exploration of the topic of model-based reinforcement learning, and propose possible future research directions.

Chapter 2

Literature Review

2.1 Background

2.1.1 Reinforcement Learning

The reinforcement learning framework consists of an agent learning from its interactions with the environment [25]. With every action a_t taken by the agent, the environment returns a state s_{t+1} and a reward r_{t+1}. Reinforcement learning problems can be formally modelled as a Markov Decision Process (MDP). A Markov Decision Process is a 4-tuple (S, A, T_a, R_a), where S is a set of states, A is a set of actions where $A_s \subseteq A$ is the set of actions available from state s. T_a is the transition function, where $T_a(s, s')$ is the probability that action a in state s at time t will lead to state s' at time $t + 1$. $R_a(s, s')$ is the immediate reward received after transitioning from state s to state s' having taken action a.

The goal of reinforcement learning is to find the optimal policy $a = \pi^*(s)$, a function that gives the best action a in all states $s \in S$. The optimal policy can be found either directly through model-free or through an environment model in model-based reinforcement learning. The goal of an MDP is to find a policy $\pi(s)$ that chooses an action in state s that will maximise the reward. The expected sum of future rewards $V^\pi(s) = E[\sum_{t=0}^{\infty} \gamma^t R_{\pi(s_t)}(s_t, s_{t+1})]$, discounted with parameter γ over t time steps, with $s = s_0$. $V^\pi(s)$ is the value function of a state. When combined with deep learning, the policy π_θ is determined by the parameters θ of the neural network.

2.1.2 Model-Based Reinforcement Learning

Deep reinforcement learning has shown to be highly sample inefficient in terms of real-world data. In many examples of offline learning, where batches of samples of interactions with the environment is first collected then used to train the algorithm offline [15], and in inverse reinforcement learning, where the agent aims to model an expert based on the expert's interactions with the environment [18], there exists limited real-world data. The limitation in samples motivate the need for model-based reinforcement learning, defined by having a model that models the environment dynamics, allowing sampling from the environment model to obtain model samples, improving real-world sample efficiency.

In model-based reinforcement learning (MBRL), a transition model $T_a(s, s')$ and possibly a reward

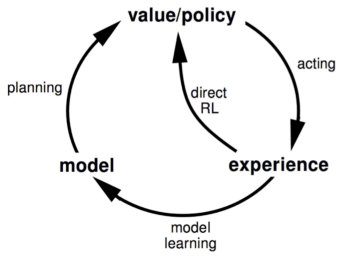

Figure 2.1: Model-Based Reinforcement Learning [25]

model $R_a(s, s')$ are learnt, for example by supervised learning. When no transition model is given by the problem, the model can be learned by sampling the environment, and be used with planning to update the policy and value. When the complexity of learning the transition/reward model is smaller than the complexity of learning the policy model directly, and planning is fast, then the model-based approach can be more sample efficient [21].

Ha et al (2018) showed that a controller can be trained using only data generated from world models, without using any real-world data [7]. Extending on world models, Hafner et al (2019) trained a reinforcement learning agent only on model data [8]. However, benchmarking research describes a dynamics bottleneck by which model-based algorithms plateau at a performance well below their model-free counterparts, showing that having more data does not necessarily result in better performance. Wang et al (2019) suggest the dynamics bottleneck can be explained by the model prediction error accumulating with time and the coupling of the policy and the learning of dynamics, making agents more prone to performance local-minima [28].

2.1.3 State-Space Models

State-space models (SSMs), such as hidden Markov models (HMMs) and recurrent neural networks (RNNs), are often used to model sequential data and the corresponding transition functions. SSMs that use deep neural networks can learn complex non-linear transitions and rich high-dimensional observations, such as images. Given a probabilistic graphical model, the joint distribution of a SSM can be factorized as:

$$p_\theta(x_{1:T}, r_{1:T}, z_{1:T}|a_{1:T}) = \Pi_{t=1}^{T} p_\theta(x_t|z_t)p_\theta(r_t|z_t)p_\theta(z_t|z_{t-1}, a_{t-1}))$$

where θ are learnable model parameters, $x_{1:T}$ denotes all observations from $t = 1, ..., T$, and likewise for $r_{1:T}$, $z_{1:T}$ and $a_{1:T}$, and the 3 distributions in the factorization correspond to observations $p_\theta(x_t|z_t)$, rewards $p_\theta(r_t|z_t)$, and transitions $p_\theta(z_t|z_{t-1}, a_{t-1})$.

The state space model can be viewed as a partially-observable Markov decision process (POMDP), where θ is learned from observed data $D = x_t, a_t, r_{t}{}_{t=1}^{T}$. Since maximum likelihood estimation is intractable as latent z_t's need to be marginalized out, the evidence lower bound (ELBO) under the data distribution p_d can be optimized, i.e. $E_{p_d}[L_e] \leq E_{p_d}[logp_\theta(x_{1:T}, r_{1:T}|a_{1:T})]$, where

$$L_e = \sum_{t=1}^{T} (E_{q_\phi(z_t)}[logp_\theta(x_t|z_t)] + E_{q_\phi(z_t)}[logp_\theta(r_t|z_t)] - E_{q_\phi(z_{t-1})}[D_{KL}[q_\phi(z_t)\|p_\theta(z_t|z_{t-1}, a_{t-1})]])$$

and q_ϕ is a variational distribution parameterized by ϕ.

Multiple works utilise state-space models in model-based reinforcement learning in two main ways, in planning and data generation, and state-space models are commonly employed in a multi-step prediction fashion. Hafner et al (2019) extends World Models to use the cross entropy method (CEM) for multi-step predictions for planning, selecting the best action from the horizon [9]. Current work in state space models in model-based reinforcement learning explore how different simulated experiences in the Dyna framework can improve agent performance [6] [20] [5]. The direction of dynamics modelling is also explored that investigates the combination of forward and backward dynamics models, showing that reduced reliance on a single forward model for dynamics modelling can reduce the prediction error over multiple time-steps, reducing compounding error over time [14] [12] [11] [1].

2.2 Motivation

Wang et al (2019) proposes a few key research challenges for future model-based reinforcement learning research, two of which are the dynamics bottleneck and the planning horizon dilemma [28]. From benchmarking experiments, these two challenges are shown to critically affect the performance of model-based reinforcement learning algorithms.

2.2.1 Dynamics Bottleneck

The dynamics bottleneck in model-based reinforcement learning, the bottleneck by which model-based reinforcement learning algorithms plateau in performance at a level well below their model-free counterparts, shows that more data available for agent training does not necessarily result in better performance. Wang et al (2019) proposes 2 assumptions can potentially explain the dynamics bottleneck. First, prediction error of the environment dynamics model accumulates with time, also known as the compounding error phenomenon [26] [2] [22]. Second, the policy and learning of dynamics objectives are coupled together, resulting in agents being more prone to performance local-minima.

The compounding error phenomenon was tackled through a few approaches, such as probabilistic ensembles proposed to capture uncertainty [3] and using multi-step prediction transition models conditioned on multiple previous time-steps [10]. However, benchmarking experiments show that prediction under probabilistic ensembles still becomes unstable and inaccurate with time [3].

The coupling of policy and dynamics learning objectives presents an objective mismatch between the reward-maximising agent and the environment model that models the transition function, resulting in the performance bottleneck below model-free methods [13]. The agent aims to maximise the reward obtained after optimizing the decision either in planning or in control, while the dynamics model maximises the log-likelihood of real-world sample data. Experiments show that the negative log-likelihood (NLL) and reward can be poorly correlated, and NLL improvements can initially improve reward but later worsen it, and furthermore, even models with similar NLLs can lead to very different rewards [13].

2.2.2 Planning Horizon Dilemma

The planning horizon is a critical hyperparameter choice in shooting methods that heavily influences model-based performance. Wang et al (2019) showed that increasing the planning horizon (i.e. increasing the number of steps in multi-step planning) does not necessarily increase agent performance, and more often instead worsens performance. They propose that the decrease in performance is a result of insufficient planning in a search space which increases exponentially with planning depth, i.e. the curse of dimensionality.

2.3 Research Directions

Multiple approaches have been proposed to address the dynamics bottleneck caused by the compounding error phenomenon and the objective mismatch problem.

To address the compounding error phenomenon by reducing model prediction error accumulating with time, probabilistic ensembles are proposed to capture uncertainty [3], random shooting over multiple time-steps reduce the compounding error effect [23], and multiple parallel lines of research in re-weighting samples using importance sampling, off-policy policy evaluation and inverse propensity weighting have been proposed [13] [27] [24].

To tackle the objective mismatch problem in model-based reinforcement learning, value-aware methods integrate information from the value function into the dynamics model training [17] [19] [4] or through re-weighting the loss of the dynamics model [13] [27] [24]. Another approach to alleviate the dynamics bottleneck is through using different simulated experiences in the Dyna framework to train the environment model, which has been shown to play a significant role in improving sample efficiency [6] [20] [5].

Therefore, the direction of this research project is towards improving multi-step model-based reinforcement learning, through alleviating the dynamics bottleneck, in context of the compounding error phenomenon and the objective mismatch problem, while in consideration of the planning horizon dilemma. Three potentially promising approaches are re-weighting samples to reduce model prediction error accumulating over time, integrating information from the value function into the dynamics model training and training the dynamics model using different simulated experiences to improve sample efficiency.

Chapter 3

Research Progress

3.1 Robotics Experiments with Robosuite

In exploration of the various methods of model-based reinforcement learning, for the purpose of obtaining hands-on experience with conducting related research experiments, I conducted robotics simulation experiments with Kaiqi Chen, a PhD student in the research lab, comparing the performance of Dreamer [8] with an experimental algorithm, Multi-Modal Mutual Information (MuMMI).

Robosuite is a simulation framework powered by the MuJoCo physics engine for the purpose of robot learning, and provides a suite of benchmark environments for reproducible research [29]. In this experiment, the Two Arm Peg-In-Hole environment was used. The Two Arm Peg-In-Hole environment consists of two robot arms placed opposite each other. The goal of the environment is to coordinate the two robot arms to insert a peg held by one robot arm into the hole of a board held by the other robot arm.

The Multi-Modal Mutual Information (MuMMI) algorithm uses a information-theoretic training loss to encourage modalities to share a common latent space, promoting robustness to missing observations. The environment produces images of different perspective viewpoints of the two robot arms, representing multiple modalities that can be utilised by MuMMI. The purpose of the experiment is to evaluate the relative performance of having a common latent representation of different modalities, in contrast to the standard reconstruction-based ELBO, which is not robust to irrelevant noise. Experiments were conducted over 1 million episodes.

Figure 3.1: Two Arm Peg-In-Hole Robosuite Environment [29]

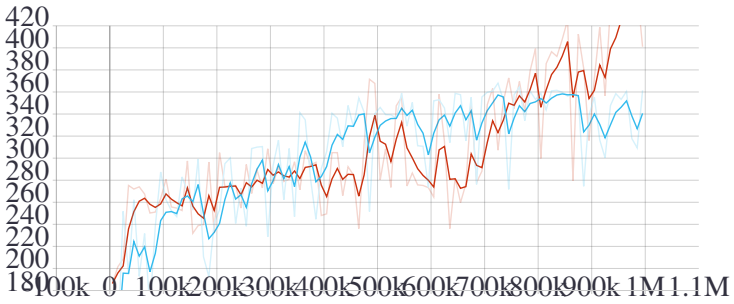

Figure 3.2: Test Return of MuMMI (blue) vs Dreamer (red), Smoothed

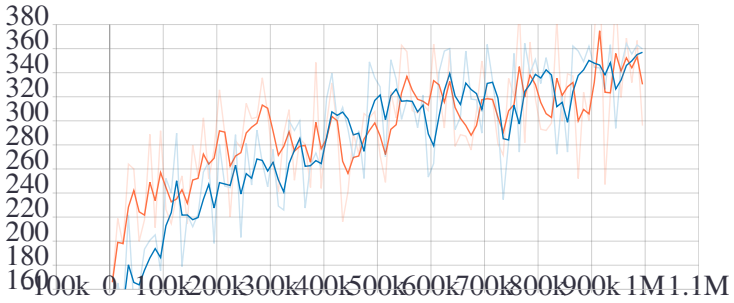

Figure 3.3: Test Return with Randomization of MuMMI(blue) vs Dreamer (orange), Smoothed

3.2 Experimental Results

3.2.1 Dreamer vs MuMMI

Comparing MuMMI against Dreamer in Figure 3.2, MuMMI's final performance on evaluation reaches around 350 while that of Dreamer reaches around 420. Given MuMMI's superior performance on other experiments in the Multi-Modal Natural MuJoCo environments, the difference suggests that the Two Arm Peg-In-Hole task is possibly too simple for the multi-modal training loss to benefit in performance.

3.2.2 Dreamer vs MuMMI with Domain Randomization

To increase the environment complexity of Two Arm Peg-In-Hole, domain randomization can be performed on the environment, increasing the necessity of relying on multiple modals of observations. Domain randomization was induced by randomization of the lighting in the environment. With lighting randomization, the test return in Figure 3.3 shows that the performance of MuMMI and Dreamer are comparable around 360, again suggesting that the environment still remains too simple for the multi-modal training performance to benefit.

The two experiments do not show that MuMMI significantly, if at all, outperforms Dreamer on the Two Arm Peg-In-Hole task. Through these experiments, it is important to note that when comparing 2 model-based reinforcement learning algorithms, the complexity of the environment must be addressed with respect to the modification made to the algorithm. In the case of MuMMI, there should exist task-relevant information that can be consolidated across multiple modals of sensors that could improve the agent performance, otherwise the difference in performance will not show.

Chapter 4

Conclusion

4.1 Future Work

4.1.1 Robotics Experiments Beyond Lighting Randomization

From this point onward, a new modification to the Two Arm Peg-In-Hole environment can be made to increase the environment complexity, in order to show the difference in capabilities between MuMMI and Dreamer. Robots can be inserted into the background of the environment performing actions unrelated to the Peg-In-Hole task. A more complex environment would mean that some modalities contain more useful information than others, and the multi-modal training loss could then show to benefit the MuMMI agent in performance.

4.1.2 Propose Potential Solutions to the Dynamics Bottleneck Problem

The dynamics bottleneck problem is a fundamental issue in model-based reinforcement learning that remains an open research problem. As discussed in section 2.3, three potential solutions are re-weighting samples to reduce model prediction error accumulating over time, value-aware methods that integrating information from the value function into the dynamics model training and the usage of different simulated experiences to improve sample efficiency.

One potential idea to be explored further through experimental validation is the combination of forward, backward and inverse dynamics models to generate different simulated experiences in a multi-step fashion to reduce the model compounding error, and then re-weighted according to the value function to address the objective mismatch problem. Therefore, I propose the following research questions to be tackled in the next semester: Can a combination of forward, backward and inverse dynamics models improve model-based reinforcement learning in terms of agent performance and real-world sample efficiency? How can the 3 directional dynamics models be integrated together to address the dynamics bottleneck? A sound method of combining multi-step dynamics models in consideration of the planning horizon dilemma, with a loss function re-weighted according to the value function, would address the dynamics bottleneck problem in model-based reinforcement learning.

References

[1] Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. *arXiv preprint arXiv:1606.07419*, 2016.

[2] Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 264–273. PMLR, 2018.

[3] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018.

[4] Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-aware loss function for model-based reinforcement learning. In *Artificial Intelligence and Statistics*, pages 1486–1494. PMLR, 2017.

[5] Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. Recall traces: Backtracking models for efficient reinforcement learning. *arXiv preprint arXiv:1804.00379*, 2018.

[6] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep q-learning with model-based acceleration. In *International Conference on Machine Learning*, pages 2829–2838. PMLR, 2016.

[7] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[8] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[9] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565. PMLR, 2019.

[10] Nan Rosemary Ke, Amanpreet Singh, Ahmed Touati, Anirudh Goyal, Yoshua Bengio, Devi Parikh, and Dhruv Batra. Learning dynamics model in reinforcement learning by incorporating the long term future. *arXiv preprint arXiv:1903.01599*, 2019.

[11] Dorothea Koert, Guilherme Maeda, Gerhard Neumann, and Jan Pcters. Learning coupled forward-inverse models with combined prediction errors. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2433–2439. IEEE, 2018.

[12] Hang Lai, Jian Shen, Weinan Zhang, and Yong Yu. Bidirectional model-based policy optimization. In *International Conference on Machine Learning*, pages 5618–5627. PMLR, 2020.

[13] Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. *arXiv preprint arXiv:2002.04523*, 2020.

[14] 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.

[15] 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.

[16] Xiao Ma, Siwei Chen, David Hsu, and Wee Sun Lee. Contrastive variational model-based reinforcement learning for complex observations. *arXiv preprint arXiv:2008.02430*, 2020.

[17] Suraj Nair, Silvio Savarese, and Chelsea Finn. Goal-aware prediction: Learning to model what matters. In *International Conference on Machine Learning*, pages 7207–7219. PMLR, 2020.

[18] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[19] Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. *arXiv preprint arXiv:1707.03497*, 2017.

[20] Yangchen Pan, Muhammad Zaheer, Adam White, Andrew Patterson, and Martha White. Organizing experience: a deeper look at replay mechanisms for sample-based planning in continuous state domains. *arXiv preprint arXiv:1806.04624*, 2018.

[21] Aske Plaat, Walter Kosters, and Mike Preuss. Model-based deep reinforcement learning for high-dimensional problems, a survey. *arXiv preprint arXiv:2008.05598*, 2020.

[22] Sébastien Racanière, Théophane Weber, David P Reichert, Lars Buesing, Arthur Guez, Danilo Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5694–5705, 2017.

[23] Anil V Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[24] Danilo J Rezende, Ivo Danihelka, George Papamakarios, Nan Rosemary Ke, Ray Jiang, Theophane Weber, Karol Gregor, Hamza Merzic, Fabio Viola, Jane Wang, et al. Causally correct partial models for reinforcement learning. *arXiv preprint arXiv:2002.02836*, 2020.

[25] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[26] Erik Talvitie. Model regularization for stable sample rollouts. In *UAI*, pages 780–789, 2014.

[27] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

[28] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[29] Yuke Zhu, Josiah Wong, Ajay Mandlekar, and Roberto Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.