Survey Paper

Model-Based Reinforcement Learning Techniques

Athul Jose P
11867566

CptS 575 Data Science
Washington State University

Contents

List of Figures

1 Introduction

Reinforcement learning (RL) algorithms are typically divided into two categories: model-free RL (MFRL) and model-based RL (MBRL). MFRL directly learns a value function or policy by interacting with the environment, while MBRL leverages these interactions to build a model of the environment. Although model-free algorithms have demonstrated success in fields such as robotics, video games, and motion animation, their high sample complexity has largely restricted their use to simulated settings. In contrast, model-based methods achieve significantly lower sample complexity by learning a model of the environment. However, accurately modeling the environment can be difficult in certain domains, and errors in the learned model often lead to ineffective policies that exploit model inaccuracies, a phenomenon known as model bias. Recent advancements have addressed the model-bias issue by incorporating probabilistic models and ensembles to quantify uncertainty in the learned models. These developments have enabled model-based methods to achieve performance comparable to model-free methods in complex domains while requiring far fewer samples.

Empirical evaluation [1] highlights three key factors that limit the performance of model-based methods. The first is the dynamics bottleneck, where algorithms that rely on learned dynamics often become stuck in local performance minima significantly below the levels achieved using ground-truth dynamics, showing no improvement even with additional data. The second is the planning horizon dilemma; while increasing the planning horizon improves the accuracy of reward estimation, it can also lead to performance drops due to the curse of dimensionality and the accumulation of modeling errors. Lastly, the early termination dilemma arises because early termination, a technique commonly used in model-free RL to encourage focused exploration and faster learning, has not yet demonstrated comparable benefits in model-based RL algorithms, thereby reducing their effectiveness in complex environments.

2 MBRL Algorithms

In this section, various MBRL present in the current literature is explained.

2.1 Model-Ensemble Trust-Region Policy Optimization (ME-TRPO)

Model-based RL can be sample-efficient but struggles with model bias and requires careful tuning, especially in complex tasks. Policies tend to exploit regions where training data is insufficient, leading to poor real-world performance. Standard techniques like regularization are ineffective against such exploitation in model-based RL. Model-Ensemble Trust-Region Policy Optimization (ME-TRPO) [2] uses and ensemble of neural networks for modeling the system dynamics. The algorithm for ME-TRPO is described on Fig. 1. Models are differentiated by weight initialization and training data sequences, capturing uncertainty and regularizing policy learning. Trust-Region Policy Optimization (TRPO) [3] is used which stabilizes learning by avoiding issues like exploding/vanishing gradients associated with back propagation through time (BPTT).

Algorithm Model Ensemble Trust Region Policy Optimization (ME-TRPO)

Initialize a policy π_θ and all models $\hat{f}_{\phi_1}, \hat{f}_{\phi_2}, ..., \hat{f}_{\phi_K}$.
Initialize an empty dataset $\mathcal{D}$.
repeat
  Collect samples from the real system f using π_θ and add them to D.
  Train all models using $\mathcal{D}$.
  repeat ▷ Optimize π_θ using all models.
    Collect fictitious samples from $\{\hat{f}_{\phi_i}\}_{i=1}^{K}$ using π_θ.
    Update the policy using TRPO on the fictitious samples.
    Estimate the performances $\hat{\eta}(\theta; \phi_i)$ for $i = 1, ..., K$.
  until the performances stop improving.
until the policy performs well in real environment f.

Figure 1: ME-TRPO Algorithm

Benefits of using ME-TRPO includes hundred times reduction in sample complexity when comparing to a model-free RL technique. It improves the stability and final performance by mitigating model bias and overfitting. ME-TRPO demonstrates superior performance on challenging continuous control tasks. The ensemble approach allows the policy to generalize well by learning to perform robustly across a range of possible scenarios generated by different model predictions. The method scales well to tasks with high-dimensional state spaces and complex dynamics, which are often intractable for traditional model-based RL techniques.

2.2 Stochastic Lower Bound Optimization (SLBO)

Stochastic Lower Bound Optimization [4] is a algorithmic framework for model-based deep reinforcement learning (RL) with theoretical guarantees, addressing challenges like sample complexity and the lack of robust theoretical understanding in non-linear dynamical systems. The SLBO algorithm depicted in Fig. 2 introduces a meta-algorithm that iteratively builds a lower bound on the expected reward using an estimated dynamic model and sampled trajectories. This lower bound is optimized jointly over the policy and the model, ensuring monotone improvement towards a local maximum of the expected reward. Notably, the approach extends the optimism-in-the-face-of-uncertainty principle to non-linear models without requiring explicit uncertainty quantification, making it applicable to complex continuous-state tasks.

Algorithm Stochastic Lower Bound Optimization (SLBO)

Initialize model network parameters ϕ and policy network parameters θ
Initialize dataset $\mathcal{D} \leftarrow \emptyset$
for n_{outer} iterations **do**
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{$ collect n_{collect} samples from real environment using π_θ with noises $\}$
  for n_{inner} iterations **do** ▷ optimize (6.2) with stochastic alternating updates
    for n_{model} iterations **do**
      optimize (6.1) over ϕ with sampled data from $\mathcal{D}$ by one step of Adam
    for n_{policy} iterations **do**
      $\mathcal{D}' \leftarrow \{$ collect n_{trpo} samples using $\widehat{M}_\phi$ as dynamics $\}$
      optimize π_θ by running TRPO on $\mathcal{D}'$

Figure 2: SLBO Algorithm

The experimental results highlight SLBO's sample efficiency, outperforming model-free algorithms like Soft Actor-Critic (SAC)[5] and TRPO while achieving comparable or superior final performance. This is attributed to the use of norm-based model loss and iterative updates alternating between the policy and the model, which reduces overfitting. The paper also addresses practical concerns, such as leveraging entropy regularization for improved exploration and robustness. However, limitations arise in guaranteeing the framework's conditions under limited data, highlighting challenges in balancing optimism and robustness.

2.3 Model-Based Meta-Policy-Optimzation (MB-MPO)

To combine the data efficiency of model-based methods with the high performance of model-free methods, Model-Based Meta-Policy-Optimzation [6] leverages meta-learning to optimize policies that are both adaptive and data-efficient. MB-MPO uses an ensemble of learned dynamics models and frames policy optimization as a meta-learning problem. The policy learns to adapt to any model in the ensemble with one gradient step, internalizing consistent predictions while accounting for discrepancies during the adaptation step. By focusing on

adaptation rather than robustness to model inaccuracies, MB-MPO mitigates model bias. This approach allows the policy to exhibit plasticity in regions of high model uncertainty and converge to optimal behaviors as models improve. Experiments demonstrate that MB-MPO matches the asymptotic performance of state-of-the-art model-free methods with up to 100× less data. It also consistently outperforms prior model-based approaches on complex control tasks, making it particularly suitable for real-world robotics applications.

Algorithm 1 MB-MPO

Require: Inner and outer step size α, β

1: Initialize the policy $\pi_{\boldsymbol{\theta}}$, the models $\hat{f}_{\boldsymbol{\phi}_1}, \hat{f}_{\boldsymbol{\phi}_2}, ..., \hat{f}_{\boldsymbol{\phi}_K}$ and $\mathcal{D} \leftarrow \emptyset$
2: **repeat**
3: Sample trajectories from the real environment with the adapted policies $\pi_{\boldsymbol{\theta}'_1}, ..., \pi_{\boldsymbol{\theta}'_K}$. Add them to $\mathcal{D}$.
4: Train all models using $\mathcal{D}$.
5: **for all** models $\hat{f}_{\boldsymbol{\phi}_k}$ **do**
6: Sample imaginary trajectories $\mathcal{T}_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using $\pi_{\boldsymbol{\theta}}$
7: Compute adapted parameters $\boldsymbol{\theta}'_k = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta})$ using trajectories $\mathcal{T}_k$
8: Sample imaginary trajectories $\mathcal{T}'_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using the adapted policy $\pi_{\boldsymbol{\theta}'_k}$
9: **end for**
10: Update $\boldsymbol{\theta} \rightarrow \boldsymbol{\theta} - \beta \frac{1}{K} \sum_k \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta}'_k)$ using the trajectories $\mathcal{T}'_k$
11: **until** the policy performs well in the real environment
12: **return** Optimal pre-update parameters $\boldsymbol{\theta}^*$

Figure 3: MB-MPO Algorithm

MB-MPO offers several advantages. Its data efficiency makes it ideal for real-world robotics tasks where collecting samples is expensive, requiring just a fraction of the data needed by model-free methods. By collecting tailored data in regions of high model uncertainty, MB-MPO improves the accuracy of learned dynamics models and accelerates policy optimization. The method's reliance on meta-learning ensures that learned policies can adapt quickly, reducing the need for extensive fine-tuning when deployed in real environments. Additionally, MB-MPO avoids the complexities of probabilistic dynamics models or parameter noise exploration, making it simpler to implement than other approaches.

Despite its successes, the paper identifies areas for future exploration. While MB-MPO uses deterministic ensembles of dynamics models, integrating Bayesian neural networks could improve uncertainty quantification. Furthermore, applying MB-MPO to real-world systems, such as robotics, is an exciting direction for extending its applicability.

2.4 Probabilistic Inference for Learning Control (PILCO)

A model based policy search framework is presented in Probabilistic Inference for Learning Control [7] which leverages probabilistic Gaussian Process (GP) models to represent system

dynamics, explicitly accounting for uncertainity. This approach reduces the effects of model errors, enabling faster learning from limited data without requiring prior task-specific knowledge, such as expert demonstrations or simulators. A key feature of PILCO is its use of a non-parametric GP model to capture the dynamics of the environment. This model provides a probabilistic understanding of transitions between states, incorporating uncertainty in the predictions. Unlike deterministic models, which can lead to unreliable long-term predictions when data is sparse, the GP model's probabilistic nature allows for a more robust estimation of future states. This uncertainty is integrated into the planning process, ensuring that the learned policies are more reliable even in challenging scenarios.

Algorithm 1 PILCO

init: Sample controller parameters $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Apply random control signals and record data.
repeat
  Learn probabilistic (GP) dynamics model, see Sec. 3.1, using all data
  repeat
    Approximate inference for policy evaluation, see Sec. 3.2: get $J^\pi(\boldsymbol{\theta})$, Eq. (9)–(11)
    Gradient-based policy improvement, see Sec. 3.3: get $\mathrm{d}J^\pi(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$, Eq. (12)–(16)
    Update parameters $\boldsymbol{\theta}$ (e.g., CG or L-BFGS).
  until convergence; **return** $\boldsymbol{\theta}^*$
  Set $\pi^* \leftarrow \pi(\boldsymbol{\theta}^*)$
  Apply π^* to system and record data
until task learned

Figure 4: PILCO Algorithm

PILCO employs analytic gradients for policy improvement, optimizing policies directly in parameter space without relying on value function approximations or state-space discretization. The use of deterministic approximate inference methods, such as moment matching, allows for efficient long-term predictions while maintaining computational feasibility. These innovations make PILCO highly data-efficient compared to other RL methods, often requiring an order of magnitude less interaction time to achieve comparable performance.

2.5 Iterative Linear Quadratic-Gaussian (iLQG)

Iterative Linear Quadratic-Gaussian [8] is an online trajectory optimization method and software platform designed for complex humanoid robots performing tasks such as getting up from arbitrary poses and recovering from disturbances using acrobatic maneuvers. The method, which computes behaviors only 7 times slower than real-time on a standard PC, is demonstrated on various tasks like the acrobot problem, planar swimming, and one-legged hopping.

The key approach is Model Predictive Control (MPC), which allows for optimal control by defining high-level goals via simple cost functions and synthesizing the corresponding behavior

and control law. MPC avoids the curse of dimensionality that limits dynamic programming by re-optimizing the trajectory and control sequence at each time step, starting from the current state. The method also uses a warm-start technique, which accelerates convergence.

While MPC is effective in slower domains, robotics faces challenges due to fast dynamics and contact phenomena. Despite these challenges, advances in the MuJoCo physics simulator (which speeds up dynamics computation), improvements to the LQG method for trajectory optimization, and a simplified model of contact dynamics have made this approach viable for dexterous robots. The authors believe that MPC will revolutionize robot control, enabling complex behaviors that were previously only seen in movies.

2.6 Stochastic Value Gradients (SVG)

Stochastic Value Gradients (SVG) [9] is a unified framework for learning continuous control policies using backpropagation, which supports both deterministic and stochastic control. The framework treats stochasticity in the Bellman equation as a deterministic function of exogenous noise, enabling the development of various policy gradient algorithms. These range from model-free methods that use value functions to model-based methods that do not require value functions.

Algorithm 1 SVG(∞)

Given empty experience database $\mathcal{D}$
for trajectory $= 0$ **to** ∞ **do**
  for $t = 0$ **to** T **do**
    Apply control $\mathbf{a} = \pi(\mathbf{s}, \eta; \theta)$, $\eta \sim \rho(\eta)$
    Insert $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$ into $\mathcal{D}$
  end for
  Train generative model $\hat{\mathbf{f}}$ using $\mathcal{D}$
  $v'_{\mathbf{s}} = 0$ (finite-horizon)
  $v'_{\theta} = 0$ (finite-horizon)
  for $t = T$ **down to** 0 **do**
    Infer $\xi|(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ and $\eta|(\mathbf{s}, \mathbf{a})$
    $v_\theta = [r_{\mathbf{a}}\pi_\theta + \gamma(v'_{\mathbf{s}'}\hat{\mathbf{f}}_{\mathbf{a}}\pi_\theta + v'_\theta)]\big|_{\eta,\xi}$
    $v_{\mathbf{s}} = [r_{\mathbf{s}} + r_{\mathbf{a}}\pi_{\mathbf{s}} + \gamma v'_{\mathbf{s}'}(\hat{\mathbf{f}}_{\mathbf{s}} + \hat{\mathbf{f}}_{\mathbf{a}}\pi_{\mathbf{s}})]\big|_{\eta,\xi}$
  end for
  Apply gradient-based update using v^0_θ
end for

Algorithm 2 SVG(1) with Replay

Given empty experience database $\mathcal{D}$
for $t = 0$ **to** ∞ **do**
  Apply control $\pi(\mathbf{s}, \eta; \theta)$, $\eta \sim \rho(\eta)$
  Observe $r, \mathbf{s}'$
  Insert $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$ into $\mathcal{D}$
  // Model and critic updates
  Train generative model $\hat{\mathbf{f}}$ using $\mathcal{D}$
  Train value function $\hat{V}$ using $\mathcal{D}$ (Alg. 4)
  // Policy update
  Sample $(\mathbf{s}^k, \mathbf{a}^k, r^k, \mathbf{s}^{k+1})$ from $\mathcal{D}$ $(k \leq t)$
  $w = \frac{p(\mathbf{a}^k|\mathbf{s}^k;\theta^t)}{p(\mathbf{a}^k|\mathbf{s}^k;\theta^k)}$
  Infer $\xi^k|(\mathbf{s}^k, \mathbf{a}^k, \mathbf{s}^{k+1})$ and $\eta^k|(\mathbf{s}^k, \mathbf{a}^k)$
  $v_\theta = w(r_{\mathbf{a}} + \gamma\hat{V}'_{\mathbf{s}'}\hat{\mathbf{f}}_{\mathbf{a}})\pi_\theta\big|_{\eta^k,\xi^k}$
  Apply gradient-based update using v_θ
end for

Figure 5: SVG Algorithm

The key contribution is the extension of value gradient algorithms to support stochastic policies in stochastic environments. This is achieved by leveraging a mathematical tool called re-parameterization, which allows for the optimization of stochastic policies. Additionally, the paper highlights that the environment dynamics model, value function, and policy can be

jointly learned using neural networks, with only observations from the environment, minimizing the impact of model errors compared to traditional methods that rely on model-predicted trajectories.

The authors propose a set of algorithms that integrate environment models with value functions to optimize policies in both deterministic and stochastic settings. These methods are applied to toy and physics-based control problems, demonstrating their effectiveness, particularly with a variant called SVG(1), which learns models, value functions, and policies simultaneously in continuous domains. The framework aims to combine the benefits of model-based and model-free approaches, reducing their respective drawbacks.

3 Conclusion

This survey highlights the strengths and challenges of model-free and model-based reinforcement learning (RL). While model-free RL excels in performance, its high sample complexity limits real-world scalability. Model-based RL, though more sample-efficient, faces challenges like model bias, the dynamics bottleneck, planning horizon dilemmas, and early termination issues. Advances such as Model-Ensemble Trust-Region Policy Optimization (ME-TRPO) have significantly improved stability, generalization, and sample efficiency, particularly in complex tasks. In summary, model-based RL shows promise in reducing sample complexity and improving applicability, but addressing its remaining challenges, such as robust uncertainty quantification and exploration strategies, is crucial for broader real-world adoption.

References

[1] T. Wang, X. Bao, I. Clavera, J. Hoang, Y. Wen, E. Langlois, S. Zhang, G. Zhang, P. Abbeel, and J. Ba, "Benchmarking model-based reinforcement learning," *arXiv preprint arXiv:1907.02057*, 2019.

[2] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel, "Model-ensemble trust-region policy optimization," *arXiv preprint arXiv:1802.10592*, 2018.

[3] J. Schulman, "Trust region policy optimization," *arXiv preprint arXiv:1502.05477*, 2015.

[4] Y. Luo, H. Xu, Y. Li, Y. Tian, T. Darrell, and T. Ma, "Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees," *arXiv preprint arXiv:1807.03858*, 2018.

[5] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[6] I. Clavera, J. Rothfuss, J. Schulman, Y. Fujita, T. Asfour, and P. Abbeel, "Model-based reinforcement learning via meta-policy optimization," in *Conference on Robot Learning*. PMLR, 2018, pp. 617–629.

[7] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 2, pp. 408–423, 2013.

[8] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4906–4913.

[9] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa, "Learning continuous control policies by stochastic value gradients," *Advances in neural information processing systems*, vol. 28, 2015.