Tsinghua University

Deep Reinforcement Learning

Lecture 7: Model-based Reinforcement Learning

Huazhe Xu

Tsinghua University

Projects

- Release ~next week!
- Feel free to work on your own idea.
- Results do not have to be positive. The projects can be open-ended. But positive results are usually more useful.
- If you really have negative results, try in-depth analysis rather than just tell everyone it does not work.

AI This Week

404 Not Found

But instead you have a quiz!
A perfect chance to have 2 bonus points!
You'll have -1 point if you did not choose it correctly.

Deep RL Quiz

诚邀您填写本问卷，扫码即可！

In Lec7

1. Model-based Planning
2. Model-based RL with learned models
3. Model-based RL with Images

In Lec7

1 Model-based Planning

2 Model-based RL with learned models

3 Model-based RL with Images

Model-Free RL

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

- Unknown transitions or dynamics
- Learning from samples

Are there scenarios we know the dynamics?

- We know the dynamics effortlessly:
  - Games: go, chess
  - Simple physic: cartpole
  - Simulated environments: Humanoid in MuJoCo simulator

- We know the dynamics with some effort:
  - System identification: spring with unknown parameters
    - Known model, unknown parameters
  - Learn the dynamics model with a statistical/math model such as a linear model or neural networks

Are these dynamics models useful?

- Yes! Why?
  - A trivial example: Running model-free RL within Atari games is an example.
  - A non-trivial example: Derive how to balance a cartpole with your physics skills.
- If we have an *exact* model of a system, what can we do?
  - Run model-free RL on it.
  - Planning or Trajectory Optimization

Objective in a Deterministic World

- a is the action, r is the reward, f is the exact dynamics model!
- Intuitively speaking
  - We know what is going to happen if we do some action.
  - Then we may calculate the cost or reward of such an action.
  - And we can think multiple steps ahead.
  - Can we find the best action sequences?
- This is very similar to how human plan to cook dinner, right?

$$\mathbf{a}_1, \ldots, \mathbf{a}_T = \arg\max_{\mathbf{a}_1, \ldots, \mathbf{a}_T} \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \text{ s.t. } \mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t)$$

Objective in a Stochastic World

- The dynamics are stochastic
- The expectation under these actions in such a stochastic world.

$$p_\theta(\mathbf{s}_1, \ldots, \mathbf{s}_T \mid \mathbf{a}_1, \ldots, \mathbf{a}_T) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$$

$$\mathbf{a}_1, \ldots, \mathbf{a}_T = \arg \max_{\mathbf{a}_1, \ldots, \mathbf{a}_T} E\left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \mid \mathbf{a}_1, \ldots, \mathbf{a}_T\right]$$

- In this world, it becomes suboptimal. Why?
  - If the future is not certain, then future information can be useful as a feedback.
  - This scenario where future is *not* used is called open loop.

Closed Loop vs Open Loop

- Open loop control:
  - Actions executed without looking at the new information
- Closed loop control:
  - Use the information (state/observation) after an action
  - For example, we may train a policy that takes in states for every timestep.
  - Another example: To balance a cartpole, we may just give a force that drag the pole back to balance position.

Open Loop Planning

$$\mathbf{a}_1, \ldots, \mathbf{a}_T = \arg\max_{\mathbf{a}_1, \ldots, \mathbf{a}_T} \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \text{ s.t. } \mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t)$$

- Simple method: Guess then check
  - Pick action sequences uniformly in the action space
  - Calculate the total rewards of each of these sequences
- This is sometimes called *random shooting*.

Can we perform better than random shooting?

- We mentioned on when we are using sampling-based method to find high Q value.
- Cross-Entropy Method (CEM)
  - Pick N action sequences from some distribution p
  - Evaluate all the action sequences
  - Choose actions based on cost/return
  - Pick top K elites, K < N
  - Update p so that it fits the K elites
- Still not good enough?
  - Curse of dimensionality
  - Open loop control

Discrete Planning Method: Monte-Carlo Tree Search

- Find the most promising leaf s_l using TreePolicy(s_1)
- Evaluate the leaf using DefaultPolicy(s_l)
- Update all values in tree between s_1 and s_l

Please read AlphaGo/AlphaZero paper to learn more: https://arxiv.org/pdf/1712.01815.pdf

Article

Mastering Atari, Go, chess and shogi by planning with a learned model

https://doi.org/10.1038/s41586-020-03051-4

Received: 3 April 2020

Accepted: 7 October 2020

Julian Schrittwieser[1,3], Ioannis Antonoglou[1,2,3], Thomas Hubert[1,3], Karen Simonyan[1], Laurent Sifre[1], Simon Schmitt[1], Arthur Guez[1], Edward Lockhart[1], Demis Hassabis[1], Thore Graepel[1,2], Timothy Lillicrap[1] & David Silver[1,2,3]✉

Trajectory Optimization with Derivatives

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_T} \sum_{t=1}^{T} c(\mathbf{x}_t, \mathbf{u}_t) \text{ s.t. } \mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$$

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_T} c(\mathbf{x}_1, \mathbf{u}_1) + c(f(\mathbf{x}_1, \mathbf{u}_1), \mathbf{u}_2) + \cdots + c(f(f(\ldots)\ldots), \mathbf{u}_T)$$

In Lec7

1 Model-based Planning

2 Model-based RL with learned models

3 Model-based RL with Images

What if the model is not known?

- Learn dynamics model from data then use what we have learned!
- Boom! Your model-based RL algorithm:

1. run base policy $\pi_0(\mathbf{a}_t \mid \mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions

- When does it work?
  - The world is very simple.
  - System Identification. If you have a great physics model.
- When does it fail?
  - In this game or near a cliff.
  - When we use a neural network!


13:45
3
Teammates
Arrow
MATYAS

Model-based RL can be improved!

1. run base policy $\pi_0(\mathbf{a}_t \mid \mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions
4. execute those actions and add the resulting data $\left\{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_j\right\}$ to $\mathcal{D}$

- Just use what you have learned to plan, e.g., MCTS!
- And of course, this planner can be a model-free RL algorithms!

Mastering Atari Games with Limited Data

Weirui Ye* **Shaohuai Liu*** **Thanard Kurutach†** **Pieter Abbeel†** **Yang Gao*‡**
*Tsinghua University, †UC Berkeley, ‡ Shanghai Qi Zhi Institute

It is somewhat open loop. Can we make it closed loop and adjust promptly?

- Model-predictive Control (MPC)

1. run base policy $\pi_0(\mathbf{a}_t \mid \mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. plan through $f(\mathbf{s}, \mathbf{a})$ to choose actions
4. execute the first planned action, observe resulting state $\mathbf{s}'$ (MPC)
5. append $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ to dataset $\mathcal{D}$

Model-based RL with a policy!

1. run base policy $\pi_0(\mathbf{a}_t \mid \mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. use $f(\mathbf{s}, \mathbf{a})$ to generate trajectories $\{\tau_i\}$ with policy $\pi_\theta(\mathbf{a} \mid \mathbf{s})$
4. use $\{\tau_i\}$ to improve $\pi_\theta(\mathbf{a} \mid \mathbf{s})$ via policy gradient
5. run $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$, appending the visited tuples $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ to $\mathcal{D}$

Why Model-based RL with a learned model?

- Data-efficiency
  - The hope is that you use little data to train model.
- Multi-task with a model
  - Re-use your world for other tasks

Why is model-based approach efficient?

Source: Sherlock Holmes

Everything looks nice, huh?

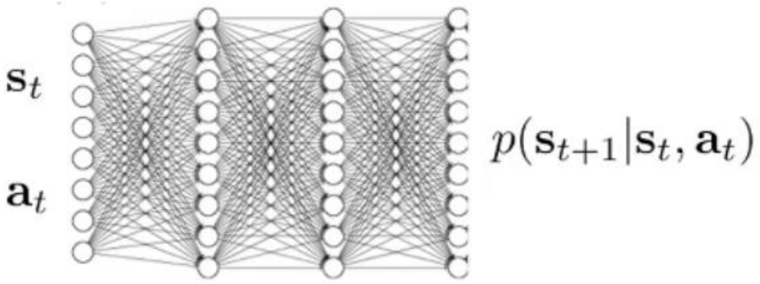
- But usually model-based algorithms can be unstable and have worse asymptotic performance.
  - Why?
  - Hint: If the model is biased toward the positive side…
  - 1. Your actions (or policies) overfit to the learned model.
  - Hint: the trajectory is really long.
  - 2. Accumulated errors.
- Any solutions?
  - To solve 1
  - To solve 2

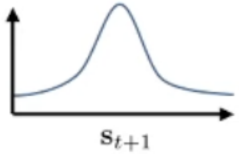
To resolve 1, uncertainty can be your friend!

- Instead of taking actions that maximize the rewards, we take actions that maximize the expected rewards.
- This might be true. But since we are touching the line between “mature knowledge” and “research stuff”. Everything can be wrong.
- I will show you later.

How to measure uncertainty?

- Can we use the output entropy?





- Is this a good measure of how uncertain the model is?
  - To answer this, we have to understand two types of uncertainties.

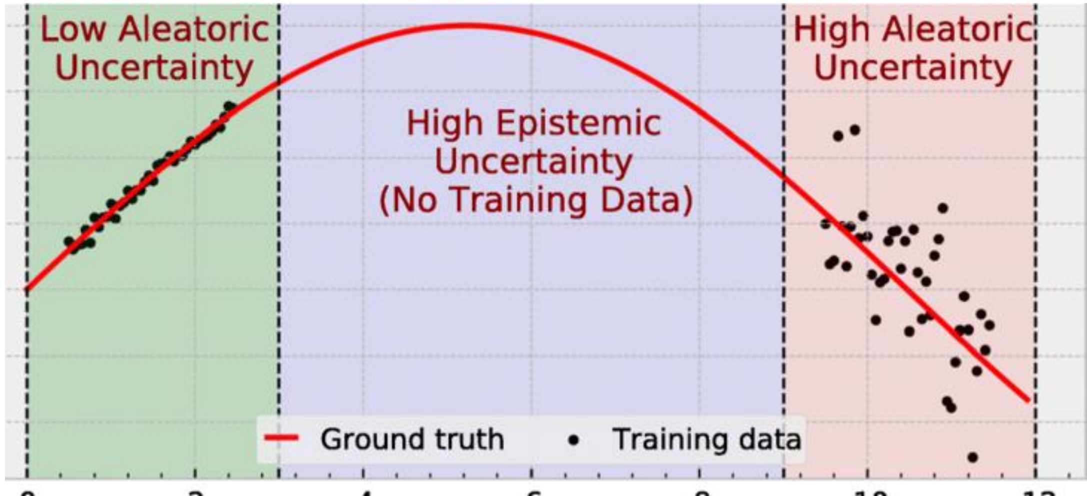
A brief introduction to the two types of uncertainty

- Aleatoric or statistical uncertainty
  - The true function itself is noisy or the innate uncertainty in the world
  - Dice
- Epistemic or model uncertainty
  - You are uncertain about the true function
- Back to our question about

output entropy

The model is certain about data,

but we are not about the model.



How to measure the uncertainty?

- We usually use the collected data to train our model.
- In other words, we want maximize $logp(D|\theta)$ by changing θ.
- Can we instead to measure $logp(\theta|D)$
- The entropy of this term is model uncertainty!
- However, this is usually intractable! Do you have some practical ideas?

Model Ensemble as an Approximation to Measure Uncertainty

- Instead of training one model
- Train multiple models
- See if they agree with each other.
- But the models have to be different in some way, right?
  - What would you do if you need to achieve this?
- Luckily, in neural nets, the randomness from initialization and SGD is strong enough to make the models different.
- But, of course, this is not the only way to measure uncertainty. If you are interested, you can try Baysian Neural Networks (https://arxiv.org/pdf/2007.06823.pdf).

Model-Ensemble MBRL

- Rough algorithm description

Step 1: sample $\theta \sim p(\theta \mid \mathcal{D})$
Step 2: at each time step t, sample $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t, \theta)$
Step 3: calculate $R = \sum_t r(\mathbf{s}_t, \mathbf{a}_t)$
Step 4: repeat steps 1 to 3 and accumulate the average reward

- The policy does not overfit to the bias of some model.

Model-Ensemble MBRL papers

MODEL-ENSEMBLE TRUST-REGION POLICY OPTIMIZATION

Thanard Kurutach **Ignasi Clavera** **Yan Duan** **Aviv Tamar** **Pieter Abbeel**
Berkeley AI Research
University of California, Berkeley
Berkeley, CA 94709
{thanard.kurutach, iclavera, rockyduan, avivt, pabbeel}@berkeley.edu

When to Trust Your Model: Model-Based Policy Optimization

Michael Janner **Justin Fu** **Marvin Zhang** **Sergey Levine**
University of California, Berkeley
{janner, justinjfu, marvin, svlevine}@eecs.berkeley.edu

Challenging the reason why model-ensemble works through a different lens!

IS MODEL ENSEMBLE NECESSARY? MODEL-BASED RL VIA A SINGLE MODEL WITH LIPSCHITZ REGULARIZED VALUE FUNCTION

Ruijie Zheng[1, §] **Xiyao Wang**[1, §] **Huazhe Xu**[2, 3] **Furong Huang**[1]
[1] University of Maryland, College Park {rzheng12, xywang, furongh}@umd.edu
[2] Tsinghua University huazhe_xu@mail.tsinghua.edu.cn
[3] Shanghai Qi Zhi Institute

- Conclusion of this paper (informally): model-ensemble works because it improves the Lipschitz condition of the value function.
- In other words, the landscape of the value function is very shaky. Ensembled model is trying to smooth it out.
- In this paper, we tried to use smoothing functions in MBRL and it works even better!

To resolve 2 (long rollouts can be error-prone), we can always use short rollouts.

1. run base policy $\pi_0(\mathbf{a}_t \mid \mathbf{s}_t)$ (e.g., random policy) to collect $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}')_i\}$
2. learn dynamics model $f(\mathbf{s}, \mathbf{a})$ to minimize $\sum_i \|f(\mathbf{s}_i, \mathbf{a}_i) - \mathbf{s}'_i\|^2$
3. pick states $\mathbf{s}_i$ from $\mathcal{D}$, use $f(\mathbf{s}, \mathbf{a})$ to make short rollouts from them
4. use both real and model data to improve $\pi_\theta(\mathbf{a} \mid \mathbf{s})$ with off-policy RL
5. run $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$, appending the visited tuples $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ to $\mathcal{D}$

An example: DYNA-style MBRL

1. collect some data, consisting of transitions (s, a, s', r)
2. learn model $\hat{p}(s' \mid s, a)$ (and optionally, $\hat{r}(s, a)$)
3. repeat K times:
4. sample $s \sim \mathcal{B}$ from buffer
5. choose action a (from $\mathcal{B}$, from π, or random)
6. simulate $s' \sim \hat{p}(s' \mid s, a)$ (and $r = \hat{r}(s, a)$)
7. train on (s, a, s', r) with model-free RL
8. (optional) take N more model-based steps

Model-Based Reinforcement Learning without *Value Equivalence*

- Learn the dynamics $M^\star$ explicitly
- Standard model-based RL algorithm:

Repeat:

1. Sample trajectories from real dynamics $M^\star$ using current policy

$$s_0 \sim D_{s_0} \quad \longrightarrow \quad s_1 \longrightarrow \quad s_2 \longrightarrow \quad s_3 \longrightarrow \quad s_4 \cdots\cdots$$

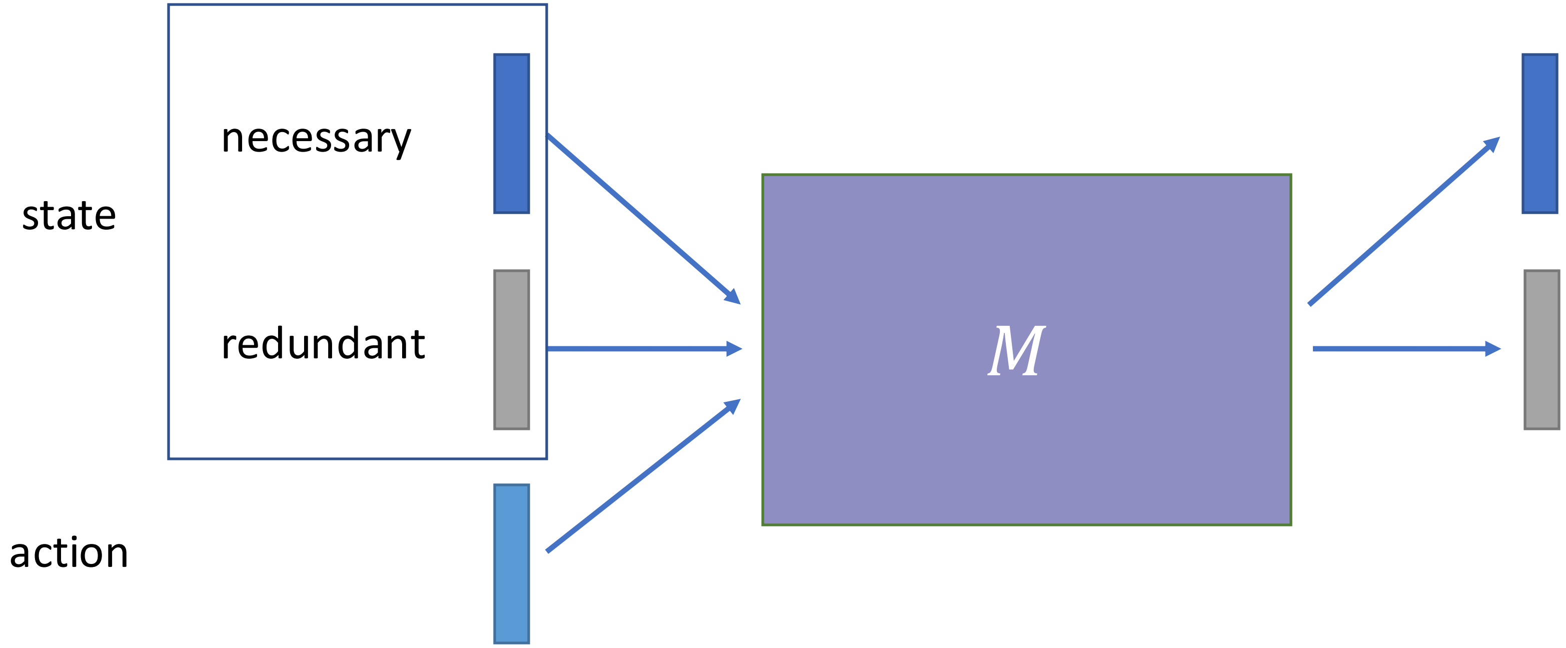
2. Learn a dynamical model using existing trajectories

$$\min_M \sum ||M(s_t, a_t) - s_{t+1}||_2^2$$

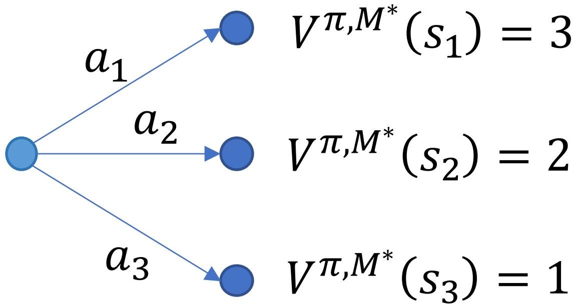
3. Find a good policy for the learned dynamics M
   - Does not cost real samples; any RL algo. may be used as a blackbox

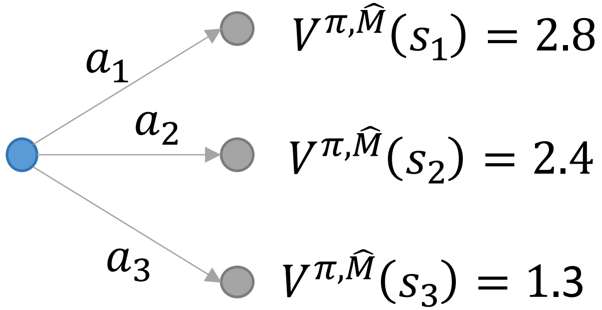
Mean-Square Error?

- not invariant to state representation!



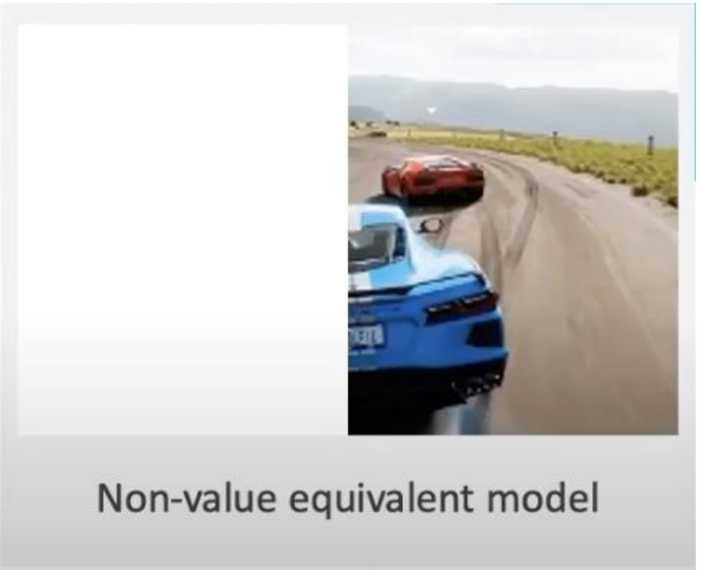
A good *model* implies a similar *value function*

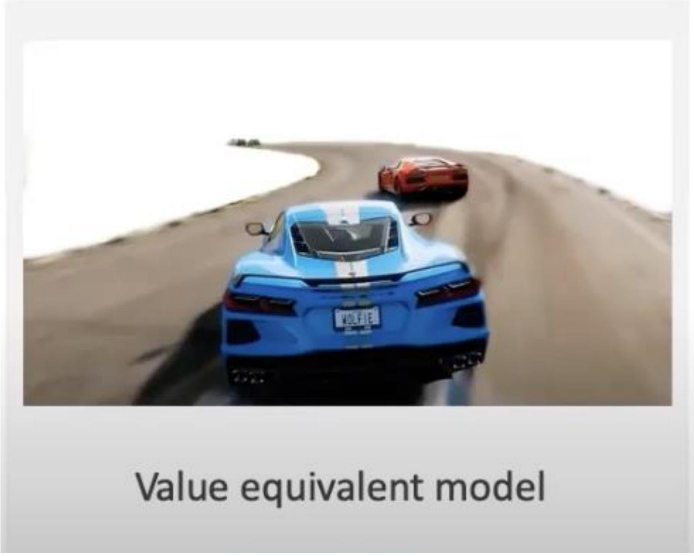




$V^{\pi,M^*}(s_i) \approx V^{\pi,M}(s_i)$ ⟹ π can generalize to M^*

An intuitive Example





Source: NiklasOPF

A new loss

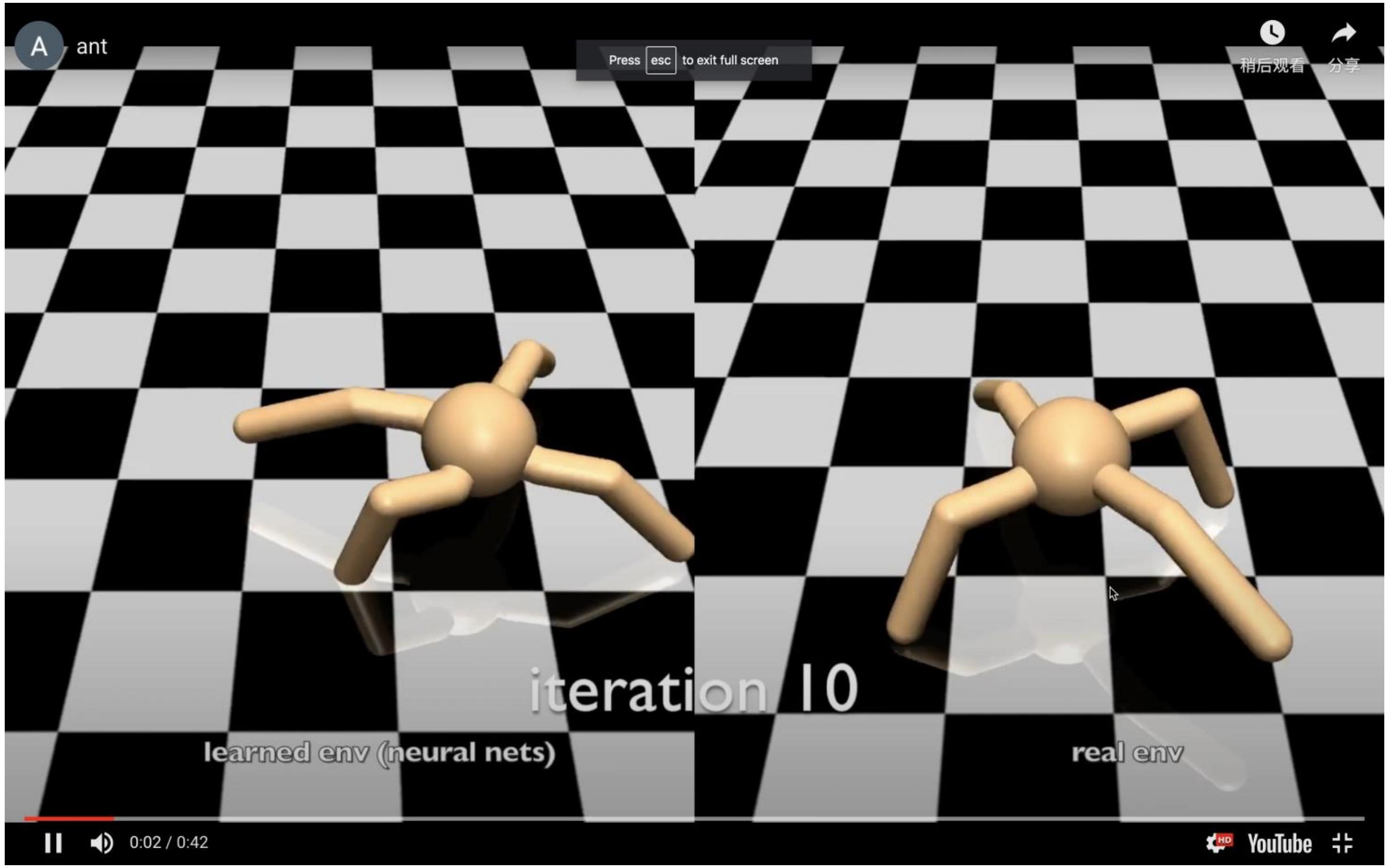
Ideal loss for $M \approx$ error of predicting future return using M

$$|V^{\pi,M} - V^{\pi,M^\star}|$$

total return on estimated dynamics M

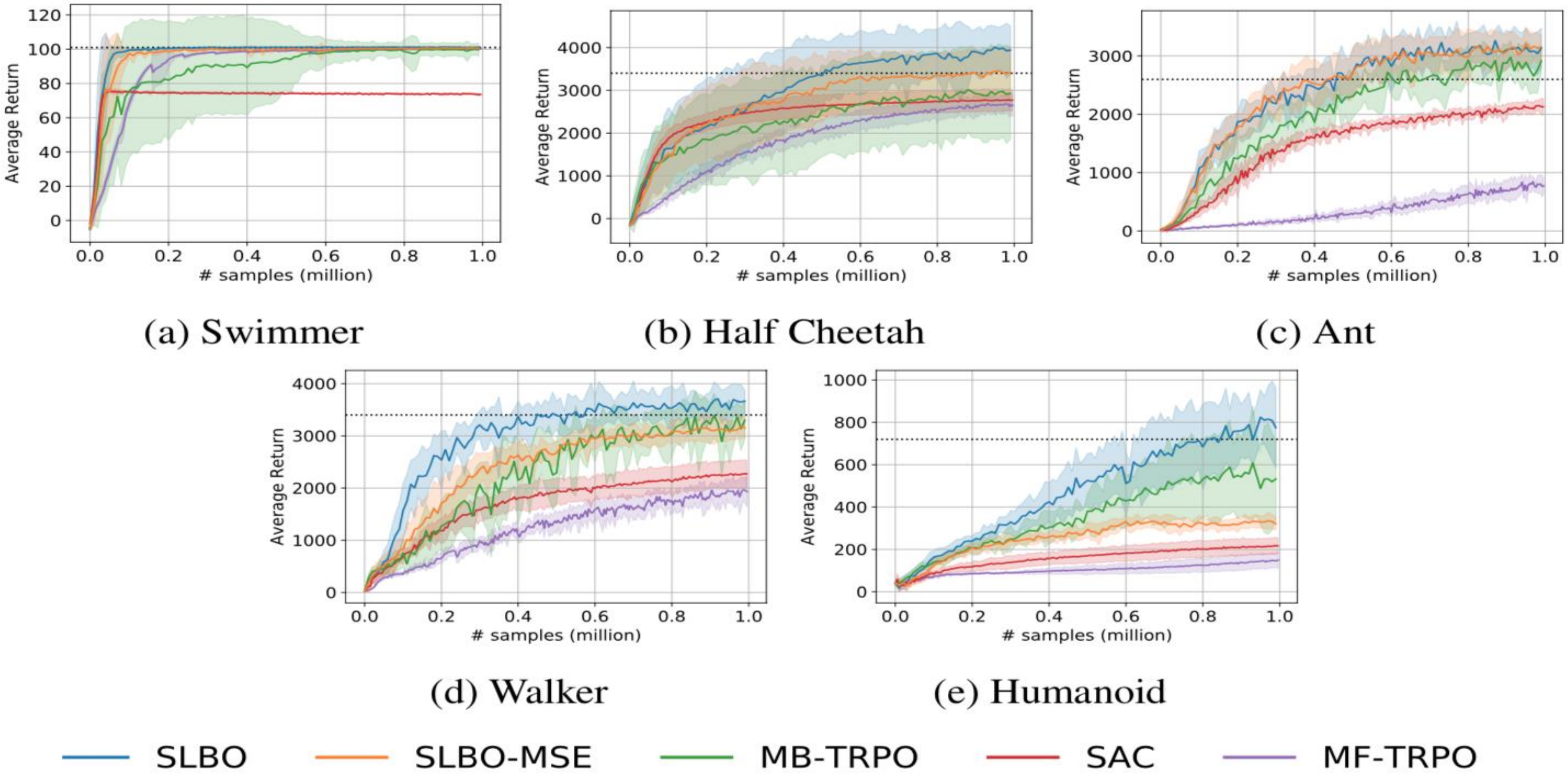
total return on true dynamics $M^\star$

Qualitative Results



Experimental Results

SOTA sample efficiency



(a) Swimmer (b) Half Cheetah (c) Ant

(d) Walker (e) Humanoid

Papers with Value Equivalence

Algorithmic Framework for Model-based Deep Reinforcement Learning with Theoretical Guarantees

Yuping Luo *[1], Huazhe Xu *[2], Yuanzhi Li[4], Yuandong Tian[3], Trevor Darrell[2], and Tengyu Ma[4]

[1]Princeton University, yupingl@cs.princeton.edu
[2]University of California, Berkeley, {huazhe_xu,trevor}@eecs.berkeley.edu
[3]Facebook AI Research, yuandong@fb.com
[4]Stanford University. {yuanzhil,tengyuma}@stanford.edu

The Value Equivalence Principle for Model-Based Reinforcement Learning

Christopher Grimm
Computer Science & Engineering
University of Michigan
crgrimm@umich.edu

André Barreto, Satinder Singh, David Silver
DeepMind
{andrebarreto,baveja,davidsilver}@google.com

Proper Value Equivalence

Christopher Grimm
Computer Science & Engineering
University of Michigan
crgrimm@umich.edu

André Barreto, Gregory Farquhar, David Silver, Satinder Singh
DeepMind
{andrebarreto,gregfar,
davidsilver,baveja}@google.com

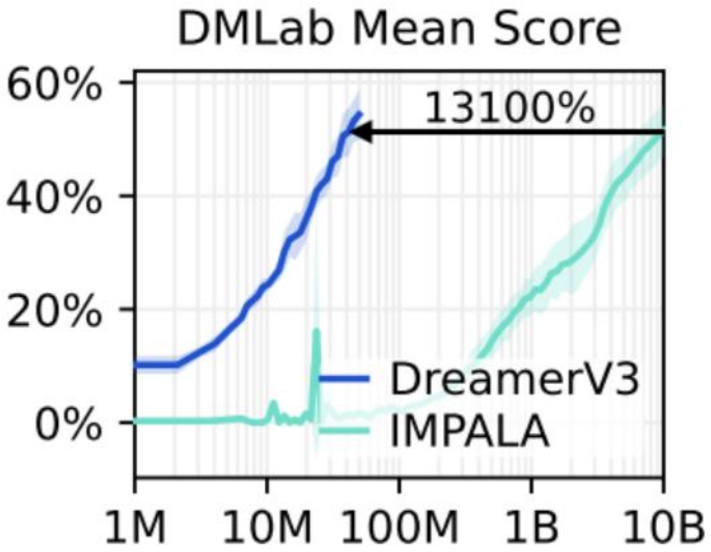
In Lec7

1. Model-based Planning
2. Model-based RL with learned models
3. Model-based RL with Images

Instead of using vector states, can MBRL deal with images?

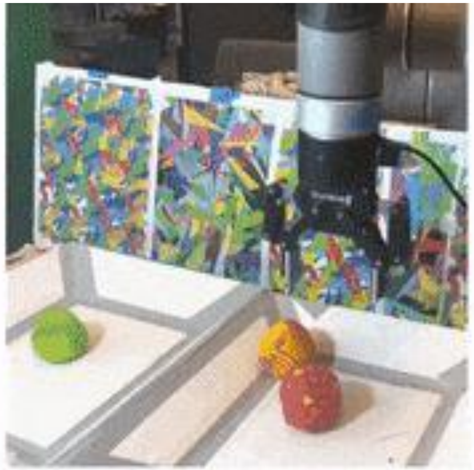
- What’s the challenge?
  - Very high-dimensional and complex
  - Redundancy
  - Partial observability
- Solutions:
  - Nothing special, use neural networks to first compress/embed the images.
  - Then predict next state and reward in the latent space.

When we touch images, many efforts are spent on how we can design the architecture!



DayDreamer Results

A1 Quadruped Walking



UR5 Multi-Object Visual Pick Place

XArm Visual Pick and Place

Sphero Ollie Visual Navigation

MBRL with Images papers & all the papers

- https://github.com/opendilab/awesome-model-based-RL

Mastering Diverse Domains through World Models

Danijar Hafner[1,2] Jurgis Pasukonis[1] Jimmy Ba[2] Timothy Lillicrap[1]

[1]DeepMind [2]University of Toronto

Temporal Difference Learning for Model Predictive Control

Nicklas Hansen, Xiaolong Wang*, Hao Su*

UC San Diego

SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J. Johnson, Sergey Levine

TD-MPC2: Scalable, Robust World Models for Continuous Control

Nicklas Hansen[*], **Hao Su**[*†], **Xiaolong Wang**[*†]
[*]University of California San Diego, [†]Equal advising
{nihansen,haosu,xiw012}@ucsd.edu

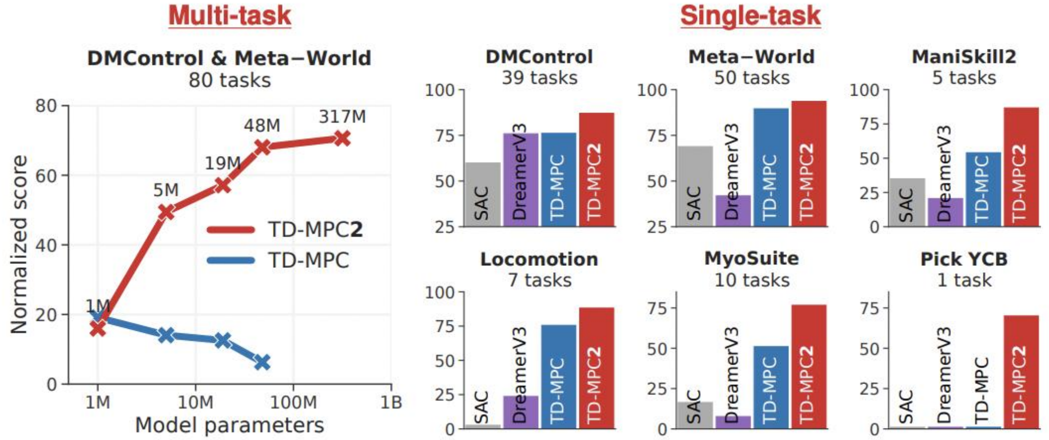
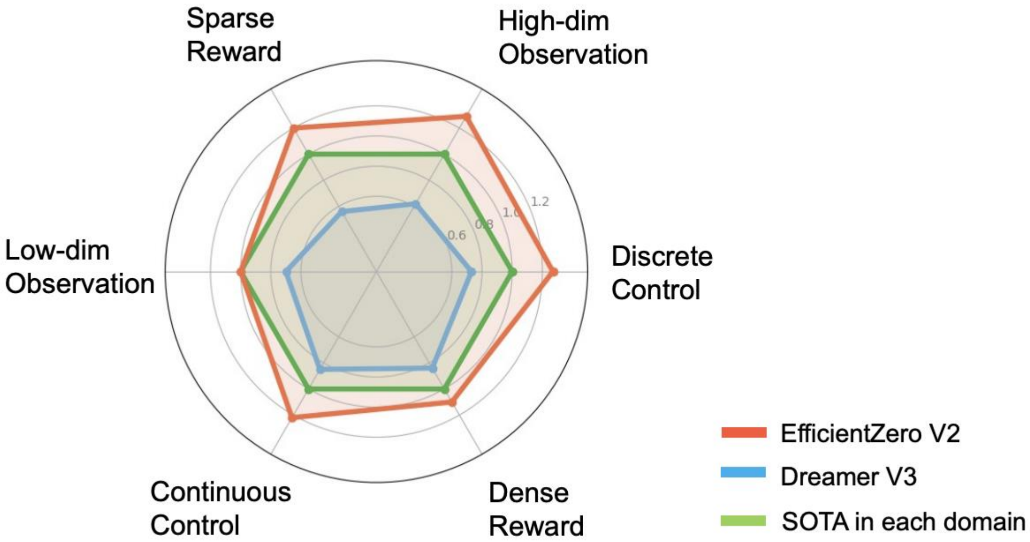


Figure 1. **Overview.** TD-MPC**2** compares favorably to existing model-free and model-based RL ... 4 continuous control tasks spanning multiple domains, with a *single* set of hyper- ... We further demonstrate the scalability of TD-MPC**2** by training a single 317M ... erform **80** tasks across multiple domains, embodiments, and action spaces (*left*).

EfficientZero V2: Mastering Discrete and Continuous Control with Limited Data

Shengjie Wang [* 1 2 3] **Shaohuai Liu** [* 1] **Weirui Ye** [* 1 2 3] **Jiacheng You** [1] **Yang Gao**[† 1 2 3]



MBRL is so good?

- Not really! There are still a lot to be improved!
- It is usually efficient in samples but slow in time.
- The multi-tasking nature is not fully explored. Many papers learn a narrow model rather than a general model.
- Given some offline data, would do learn policies from them or would you learn a model? It is not determined yet! Maybe a nice course project idea not on the list?
- Any other ideas?

MF comes back!

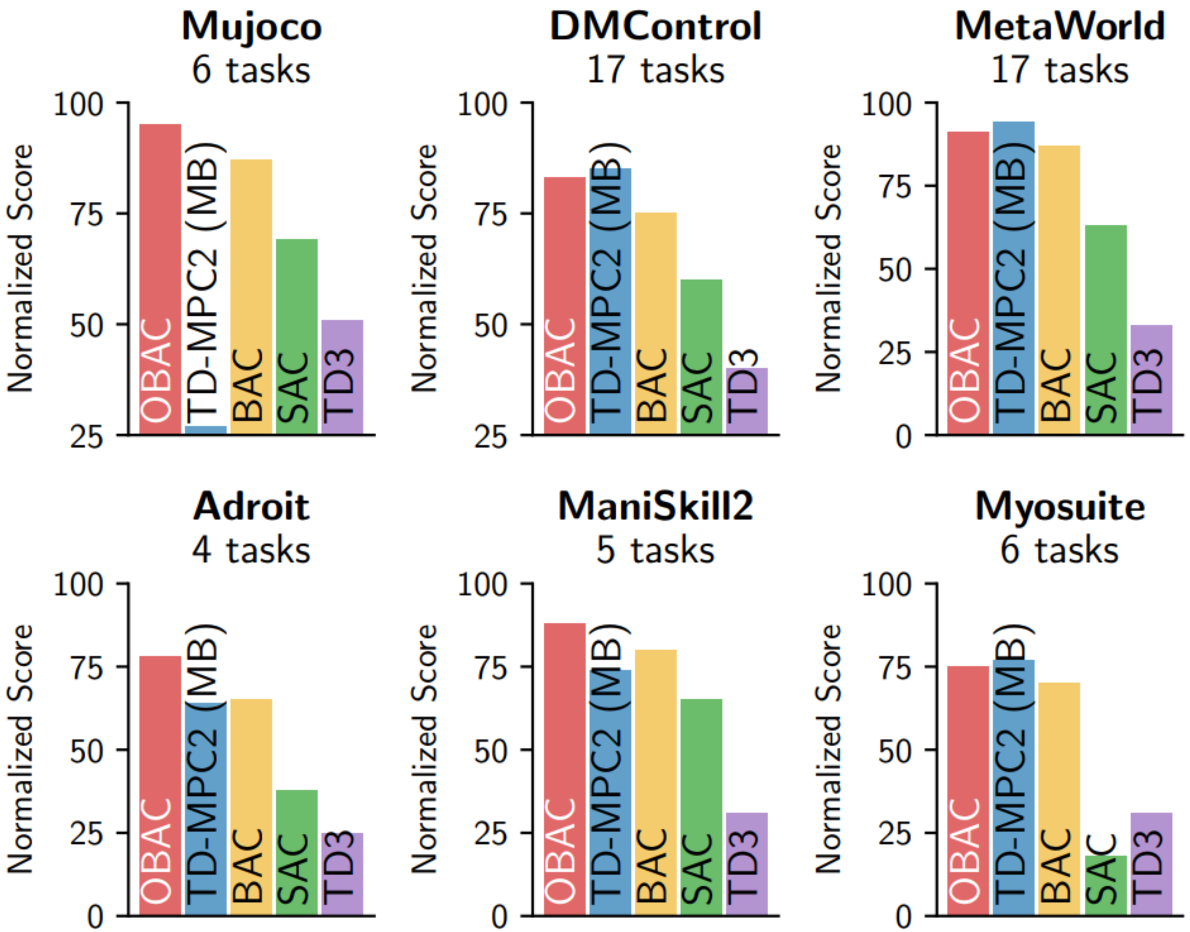


Figure 1. **Overview**. *(Top)*: we illustrate the framework of OBAC, where the concurrent offline optimal policy can boost the online learning policy with an adaptive constraint mechanism. *(Bottom)*: comparison of normalized score. Our OBAC can be comparable with advanced model-based RL method TD-MPCs, and outperform several popular model-free RL methods BAC, SAC and TD3.

Thank you!