On the Statistical Complexity of Reinforcement Learning

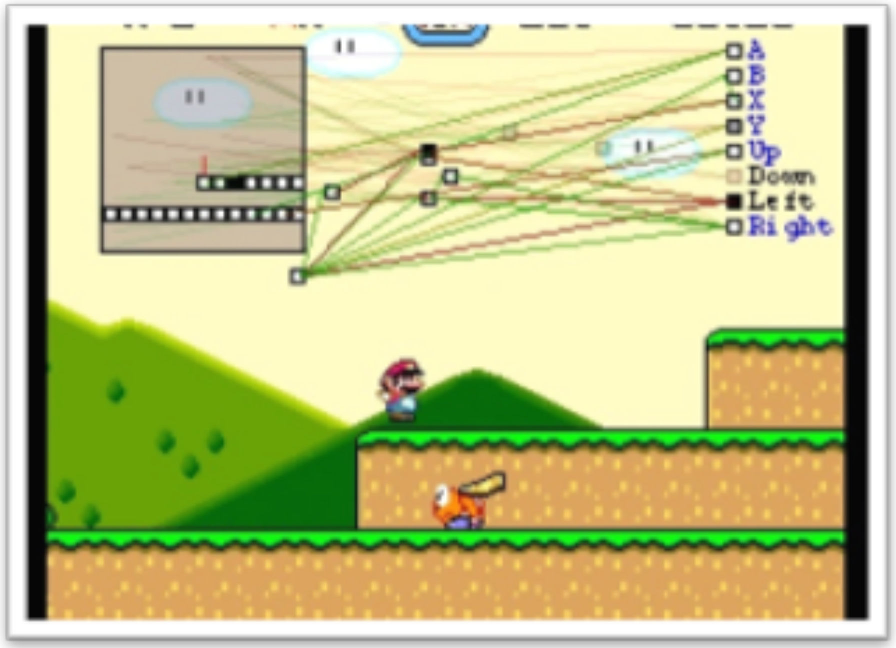
and the use of regression

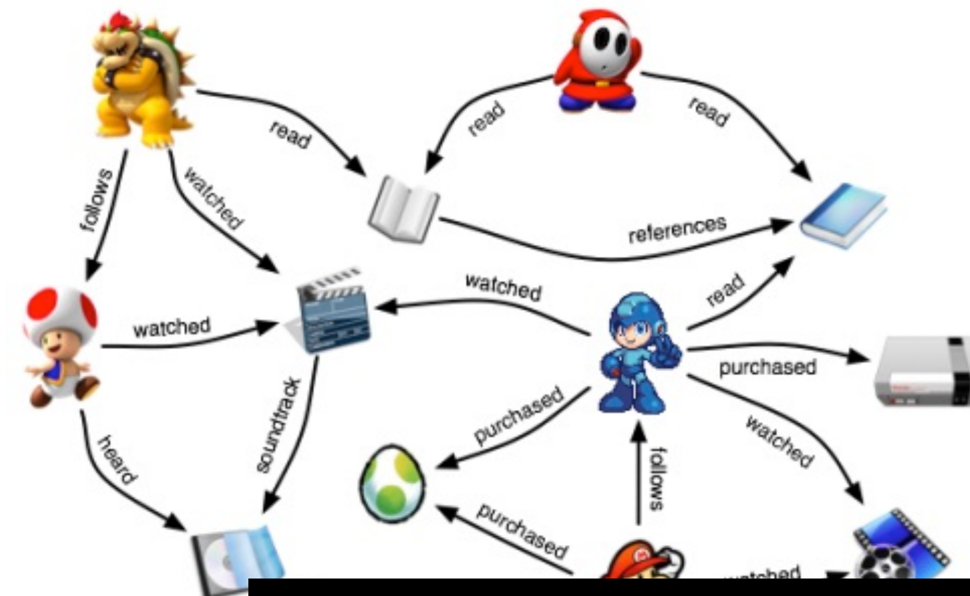
Joint work with Lin Yang, Yaqi Duan, Csaba Szepesvari, Zeyu Jia

Mengdi Wang

PRINCETON UNIVERSITY

DeepMind





Reinforcement learning achieves phenomenal empirical successes

What if data is limited?

Suppose we are given a generative model (Kakade 2003), which can sample transitions from any given user-specified state-action pair (s, a)

How many samples are necessary and sufficient to learn a 90%-optimal policy?

Tabular Markov decision process

- A finite set of states S
- A finite set of actions A
- Reward is given at each state-action pair (s,a):

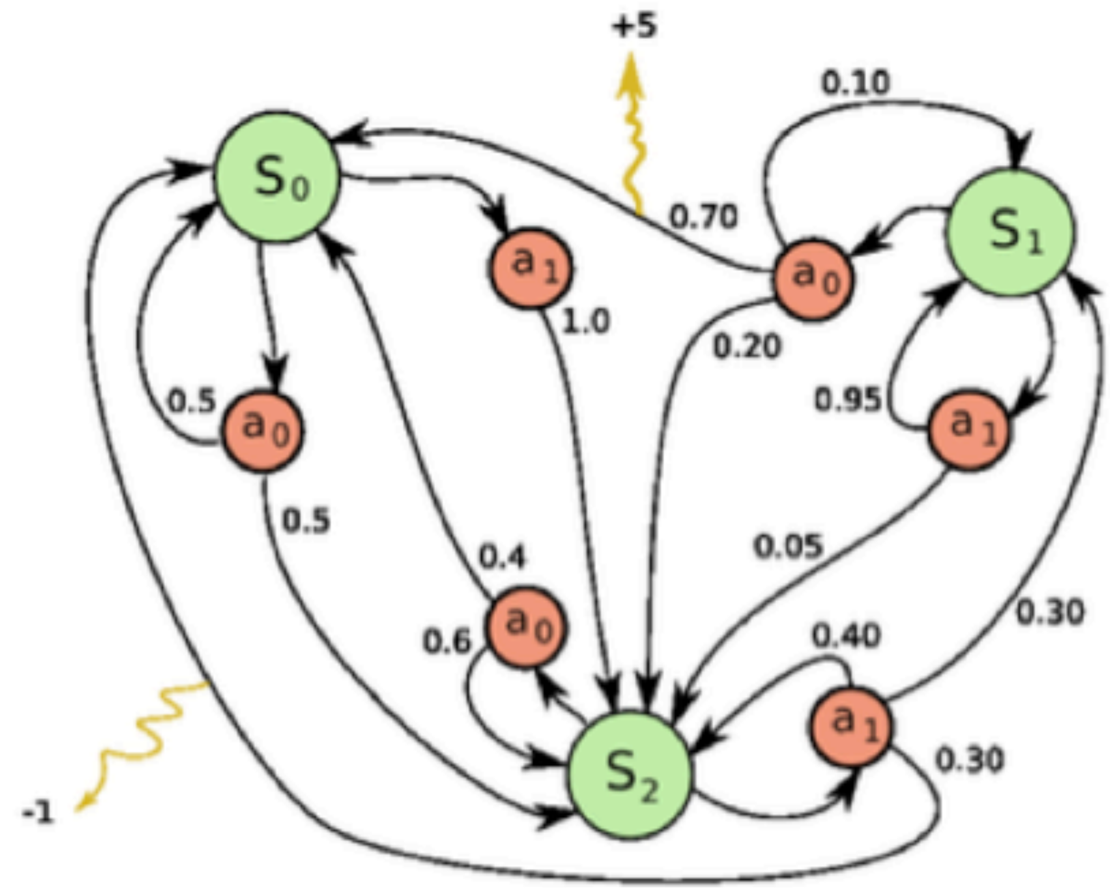
  $r(s,a) \in [0,1]$
- State transits to s' with prob.

  $P(s'|s,a)$

- Find a best policy $\pi: S \to A$ such that

$$\max_{\pi} v^{\pi} = \mathbb{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$$

- *$\gamma \in (0,1)$ is a discount factor*



(figure from google)

We call if "tabular MDP" if there is no structural knowledge at all

Prior efforts: algorithms and sample complexity results

Algorithm	Sample Complexity	References
Phased Q-Learning	$\tilde{O}(C\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^7\epsilon^2})$	[KS99]
Empirical QVI	$\tilde{O}(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^5\epsilon^2})$ [2]	[AMK13]
Empirical QVI	$\tilde{O}(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^3\epsilon^2})$ if $\epsilon = \tilde{O}(\frac{1}{\sqrt{(1-\gamma)\lvert\mathcal{S}\rvert}})$	[AMK13]
Randomized Primal-Dual Method	$\tilde{O}(C\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^4\epsilon^2})$	[Wan17]
Sublinear Randomized Value Iteration	$\tilde{O}\left(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^4\epsilon^2}\right)$	[SWWY18]
Sublinear Randomized QVI	$\tilde{O}\left(\frac{\lvert\mathcal{S}\rvert\lvert\mathcal{A}\rvert}{(1-\gamma)^3\epsilon^2}\right)$	This Paper

$1/(1-\gamma)=1+\gamma+\gamma^2+\ldots$ is the effective horizon

If $\gamma=0.99$, the speedup is 10^8 times

Minimax-optimal sample complexity of tabular MDP

- Suppose we are given a **generative model** that can sample transitions from any given (s,a)

- **Information-theoretical limit** (Azar et al. 2013)**:** Any method finding an ε-optimal policy with probability 2/3 needs at least sample size

$$\Omega\left(\frac{|SA|}{(1-\gamma)^3\epsilon^2}\right)$$

- **The optimal sampling-based algorithm** (Sidford, Wang, Yang, Ye, 2018, Agarwal et al, 2019)**:** With a generative model, finding ε-optimal policy with probability 1-δ using sample size

$$O\left(\frac{|SA|}{(1-\gamma)^3\epsilon^2}\log\frac{1}{\delta}\right)$$

S is way too big

Suppose states are vectors of dimension **d**

Vanilla discretization of state space gives **$|S| = 2^d$**

Size of policy space = **$|A|^{|S|}$**

Log of policy space size = **$|S| \log(|A|) > 2^d$**

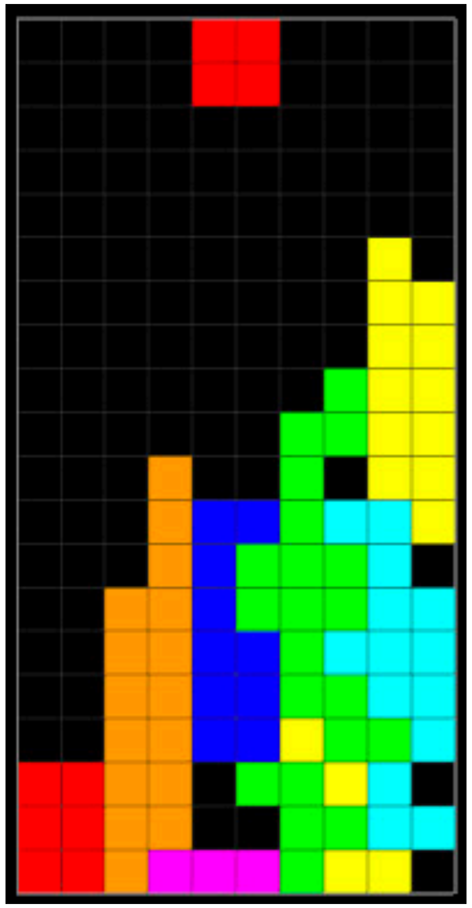
When can we solve RL provably using smaller data size ?

Adding some structure: state feature map

- Suppose we have a state feature map

$$state \mapsto [\phi_1(state), \ldots, \phi_d(state)] \in \mathbb{R}^d$$

- Now can we do better?
- Example : tetris can be solved well using 22 hand-picked features and linear models (Bertsekas & Loffe 96)
  - Feature 1: Height of wall; Feature 2: Number of holes
- Example: Neural representation trained by state-to-state regression
- Example: space lifting by random features + low-rank truncation to get low-dim state representations (with Sun 2019)



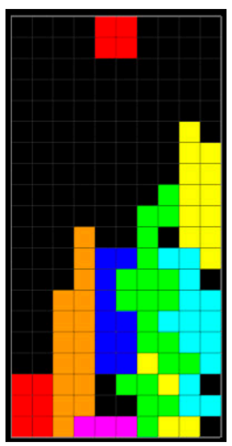
Representing value function using linear combination of features

- The value function of a policy is the expected cumulative reward as the initial state varies:

$$V^\pi : \mathcal{S} \to \mathbb{R}, \qquad V^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=0}^{H} r(s_t, a_t) \mid s_0 = s \right]$$

- Suppose that the high-dimensional value vector admits a linear model:

$$V^\pi(s) \approx w_1 \phi_1(s) + \ldots + w_N \phi_N(s)$$

- Value of  = w_1 x Height of Wall + w_2 x # Holes + …

- Let $\mathbf{H}_\phi$ be the space of value function approximators

Rethinking Bellman equation

Bellman equation is the optimality principal for MDP (in the average-reward case, where γ=1)

$$\bar{v}^* + v^*(s) = \max_a \left\{ \sum_{s' \in \mathcal{S}} P_a(s,s')v^*(s') + r_a(s) \right\}, \qquad \forall s \in \mathcal{S}$$

- The max operation applies to every state-action pair -> nonlinearity + high dim

Bellman equation is equivalent to a bilinear saddle point problem (Wang 2017)

$$\min_{v} \max_{\mu \in \Delta} \left\{ L(v,\mu) = \sum_a \left(\mu_a^T((I - P_a)v + r_a) \right) \right\}$$

value function stationary state-action distribution

- Strong duality between value function and invariant measure
- SA x S linear program
- Approximate linear programming methods for RL (Farias & Van Roy 2003)

Reducing Bellman equation using features

$$\bar{v}^* + v^*(s) = \max_a \left\{ \sum_{s' \in \mathcal{S}} P_a(s,s')v^*(s') + r_a(s) \right\}, \forall s$$

Bellman eq:

- High-dim
- Nonlinear

$$\Updownarrow$$

$$\min_{v} \max_{\mu \in \Delta} \left\{ L(v,\mu) = \sum_a \left(\mu_a^T((I - P_a)v + r_a) \right) \right\}$$

Bellman saddle point:

- High-dim

$$\Downarrow$$

$$\min_{v \in \mathbf{H}_\phi} \max_{\mu \in \mathbf{H}_\phi \times \mathbf{H}_\psi} L(v,\mu)$$

Function approximation

$$v(\cdot) \approx \sum_{i=1}^{r_S} w_i \phi_i(\cdot)$$

$$\mu(s,a) \approx \sum_{i=1}^{r_S} \sum_{j=1}^{r_A} u_{ij} \phi_i(s) \psi_j(a)$$

$$\Updownarrow$$

$$\min_{w \in \mathfrak{R}^{d_S}} \max_{u \in \mathfrak{R}^{d_A}} L(v_w, \mu_u)$$

- Low-dim
- Convex-concave
- Strong duality
- Parametric

Sample complexity of RL with features

Suppose that good state and action features are known and under the generative model:

- For average-reward MDP, a primal-dual policy learning method finds the optimal policy using sample size (with Chen, Li, 2018)

$$\Theta\left(t_{mix}^2 \tau^2 \cdot \frac{|d_S d_A|}{\epsilon^2}\right)$$

where t_{mix} is the worst-case mixing time, and τ is a constant measuring the uniform ergodicity of the MDP.

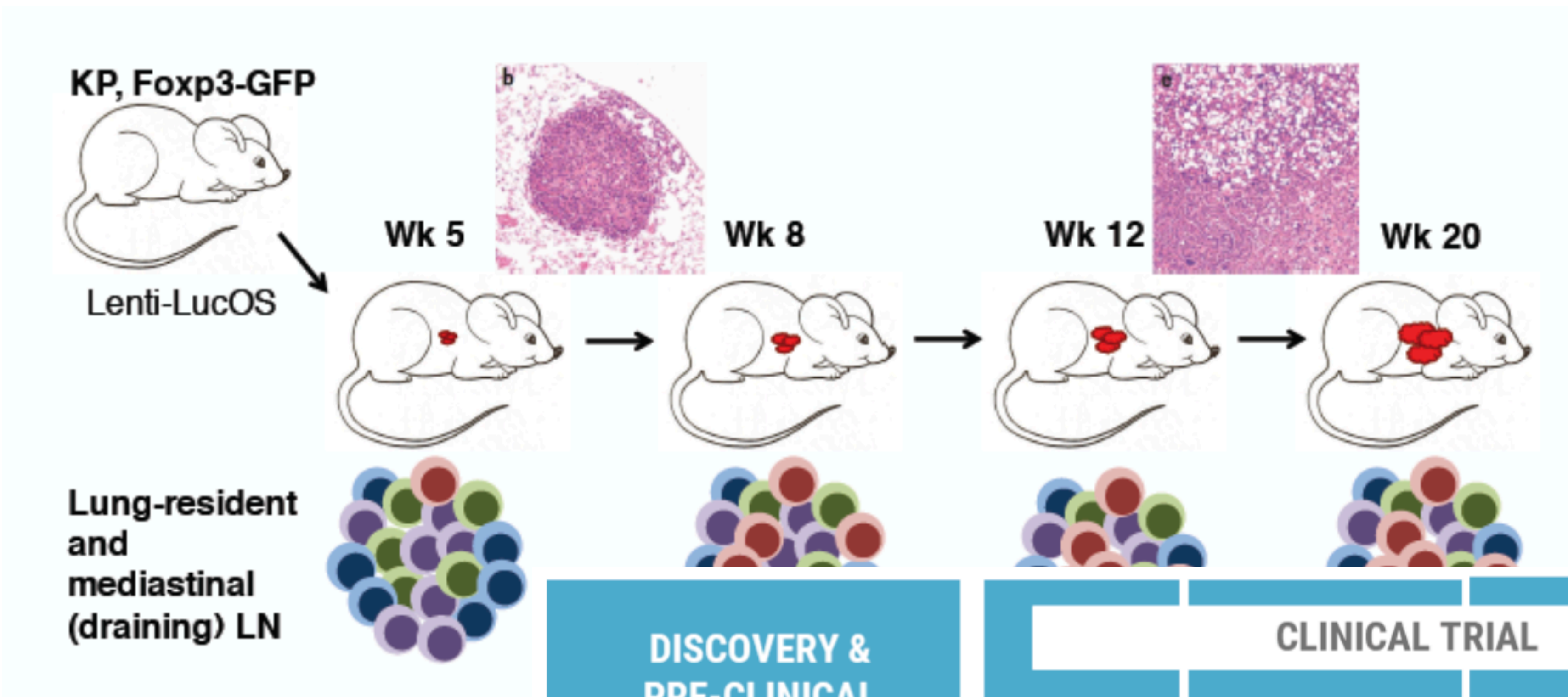
- For discounted MDP, can achieve the minimax-optimal sample complexity (with Yang, 2019)

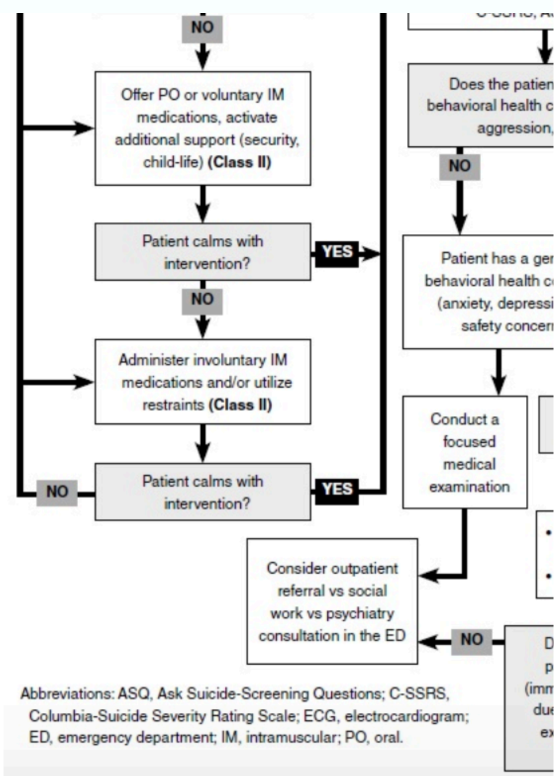
$$\Theta\left(\frac{|d_S d_A|}{\epsilon^2 (1-\gamma)^3}\right)$$

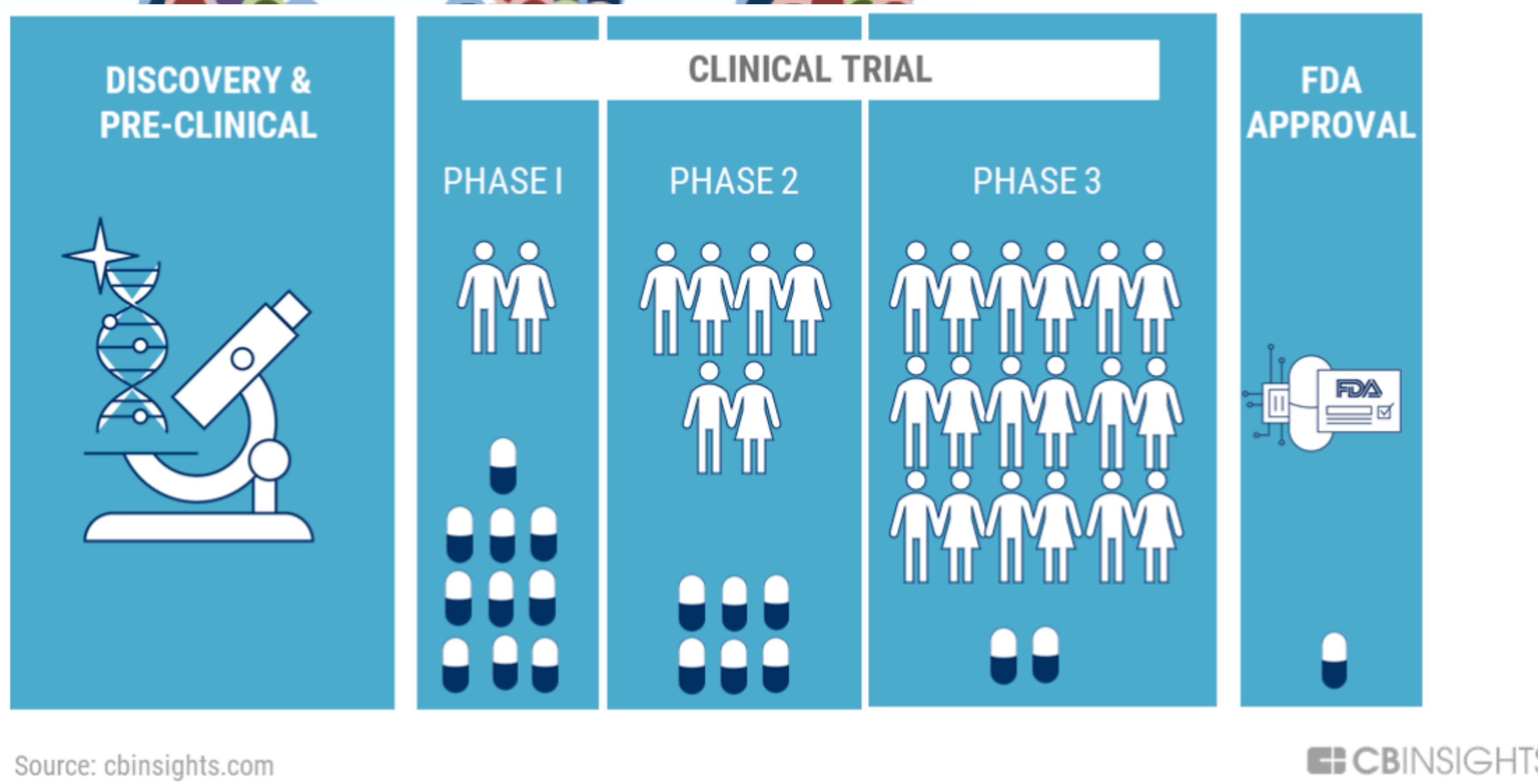
- Having good features allow us to extrapolate values from seen states to unseen states.

- Reduced sample complexity's dependence on **SA** to **d_S d_A**

How to optimally predict the performance of a new policy from past experiences?







If the data/trial is limited and costly, we have to do our best with batch data.

Off-Policy Policy Evaluation (OPE)

- Suppose we are given a dataset of state-action transitions $\mathscr{D} = \{(s, a, r', s)\}$, collected from independent H-horizon episodes

- The goal is to estimate the cumulative rewards to be earned by a target policy π from a fixed initiation distribution ξ_0:

$$v^\pi := \mathbb{E}^\pi \left[\sum_{h=0}^{H} r(s_h, a_h) \middle| s_0 \sim \xi_0 \right]$$

- Behavioral policies $\bar{\pi}$, reward r and transition functions $p(s' \mid s, a)$ are *all unknown*

OPE is a first-order task of batch data RL:

- *It is critical to data-limited applications, for examples,* predicting the effect of a new medical treatment; evaluating a new trading strategy

- *It enables downstream tasks*, such as policy improvement and continued exploration

- Existing OPE methods mainly use **importance sampling**
  - Reweighing samples according to the new policy: estimate $r^{\pi}(s)$ by averaging $\frac{\pi(a \mid s)}{\bar{\pi}(a \mid s)} r'$
  - Some variants require estimating the density ratio $\frac{\mu^{\pi}(s,a)}{\mu^{\bar{\pi}}(s,a)}$ for all (s,a)
  - Often requires knowledge of $\bar{\pi}$ or has to estimate it
  - Lots of prior efforts to analyze and improve importance sampling OPE methods (Precup, 2000) (Jiang & Li, 2016; Thomas & Brunskill, 2016). Liu et al. (2018) , Nachum et al. (2019), Dann et al. (2019), Xie et al. (2019), Yin & Wang (2020)

Challenges:

- Large error bounds for tabular MDP
- Curse of horizon - algorithms easily diverge due to explosive error accumulation
- Lack theory and solution beyond tabular MDP

OPE with function approximation

- **Assumption:** Denote the transition operator as $\mathbf{P}^\pi f = \mathbb{E}^\pi[f(s', a') \mid s, a]$. Suppose we are given a function class $\mathcal{Q}$ that is sufficiently expressive, i.e.,

$$r \in \mathcal{Q}, \qquad \mathbf{P}^\pi f \in \mathcal{Q}, \text{ if } f \in \mathcal{Q}$$

Under this assumption, the Q functions associated with the target policy π all belong to $\mathcal{Q}$

- **A direct regression approach (Fitted Q-Iteration):**

1. Estimate Q functions by iterative regression

$$\widehat{Q}^\pi_{H+1} \leftarrow 0; \qquad \text{For } h = H-1, \ldots, 0:$$

$$\widehat{Q}^\pi_h \leftarrow \arg\min_{f \in \mathcal{Q}} \left\{ \sum_{n=1}^{N} \left(f(s_n, a_n) - r'_n - \int_{\mathcal{A}} \widehat{Q}^\pi_{h+1}(s'_n, a) \pi(a \mid s'_n) \mathrm{d}a \right)^2 + \lambda \rho(f) \right\}$$

2. Estimate the policy value by

$$\widehat{v}^\pi_{\mathrm{FQI}} := \int_{\mathcal{S} \times \mathcal{A}} \widehat{Q}^\pi_0(s, a) \xi_0(s) \pi(a \mid s) \mathrm{d}s \mathrm{d}a$$

Equivalence to plug-in estimation

A Plug-In Estimator

1. Estimate the transition operator and reward function by
$$\widehat{P}^{\pi} : f \mapsto \arg\min_{g \in \mathcal{Q}} \left\{ \sum_{n=1}^{N} \left(g(s_n, a_n) - \int_{\mathcal{A}} f(s_n', a)\pi(a \mid s_n')\mathrm{d}a \right)^2 + \lambda\rho(g) \right\}.$$
$$\hat{r} := \arg\min_{f \in \mathcal{Q}} \left\{ \sum_{n=1}^{N} \left(f(s_n, a_n) - r_n' \right)^2 + \lambda\rho(f) \right\}$$

2. Estimate Q functions by $\widehat{Q}^{\pi}_{H+1} = 0, \qquad \widehat{Q}^{\pi}_{h-1} := \hat{r} + \widehat{P}^{\pi}\widehat{Q}^{\pi}_{h}, \qquad h = H, \ldots, 1.$

3. Evaluate the policy by $\hat{v}^{\pi}_{\text{Plug-in}} := \int_{s,a} \widehat{Q}^{\pi}_{0}(s,a)\xi_0(s)\pi(a \mid s)\mathrm{d}s\mathrm{d}a$

- **Theorem:** $\hat{v}^{\pi}_{\text{FQI}} = \hat{v}^{\pi}_{\text{Plug-in}}$

- In the case where $\mathcal{Q} = \{\phi(\cdot)^T w \mid w \in \mathbb{R}^d\}$, the estimated $\widehat{P}^{\pi}$ corresponds to the empirical transition kernel
$$\hat{p}(\cdot \mid s, a) := \phi(s,a)^{\top} \left(\lambda I + \sum_{n=1}^{N} \phi(s_n, a_n)\phi(s_n, a_n)^{\top} \right)^{-1} \left(\sum_{n=1}^{N} \phi(s_n, a_n)\delta_{s_n'}(\cdot) \right)$$

- One can compute the plug-in estimator using *any* MDP algorithm

Minimax-optimal batch policy evaluation

- **Theorem** (with Duan 2020)**:** The plug-in policy evaluator achieves the near-optimal error

$$\inf_{\hat{v}^\pi} \sup_{(p,\bar{\pi})} | \widehat{v}^\pi - v^\pi | \asymp H^2 \sqrt{\frac{1 + \chi^2_{\mathcal{Q}}(\mu^\pi, \bar{\mu})}{N}} + o(1/\sqrt{N})$$

where $\chi^2_{\mathcal{Q}}(p_1, p_2) := \sup_{f \in \mathcal{Q}} \frac{\mathbb{E}_{p_1}[f(x)]^2}{\mathbb{E}_{p_2}[f(x)^2]} - 1$, as long as $N \geq \Theta(dH^{3.5})$,

μ^π is the weighted (discounted) state-action occupancy measure under policy π, $\bar{\mu}$ is the occupancy measure of data

Message: regression and plug-in is efficient for off-policy evaluation

Minimax-optimal batch policy evaluation

- **$\chi^2_{\mathcal{Q}}(p_1, p_2)$ - a variant of chi-square divergence restricted to the $\mathcal{Q}$ class.** It measures the distributional mismatch that is only relevant to $\mathcal{Q}$.

- In the tabular case, $\chi^2_{\mathcal{Q}} = \chi^2$ reduces to the typical Pearson chi-square divergence

- In the case of linear function approximation $\mathcal{Q} = \{\phi(\cdot)^T w \,|\, w \in \mathbb{R}^d\}$,

$$\chi^2_{\mathcal{Q}}(p_1, p_2) \leq \text{cond}(\Sigma_1^{1/2}\Sigma_2^{-1/2})$$

is a form of relative condition number of covariance matrices

- When we have a well-behaved function class, $\chi^2_{\mathcal{Q}}(\mu^\pi, \bar{\mu})$ could be small regardless of $|\mathcal{S}|$

Lower Bound Analysis

- **Key idea: Construct an undistinguishable instance with the largest value gap**
- Given an MDP instance with transition kernel p, construct a similar instance $\tilde{p}(s' \mid s, a) = p(s' \mid s, a) + \phi(s, a)^\top \mathbf{x} \cdot q(s)$
- **Likelihood test:** Show that with high probability:

$$\log \frac{\widetilde{\mathscr{L}}(\mathscr{D})}{\mathscr{L}(\mathscr{D})} = \log \prod_{n=1}^{N} \frac{\widetilde{p}(s_n' \mid s_n, a_n)}{p(s_n' \mid s_n, a_n)} \gtrsim -\sqrt{N}\sqrt{\mathbf{x}^\top \Sigma \mathbf{x}} - N \cdot \mathbf{x}^\top \Sigma \mathbf{x}$$

- Then as long as $\sqrt{\mathbf{x}^\top \Sigma \mathbf{x}} \lesssim N^{-1/2}$, we have $\mathbb{P}\left(\frac{\widetilde{\mathscr{L}}(\mathscr{D})}{\mathscr{L}(\mathscr{D})} \geq \frac{1}{2}\right) \geq \frac{1}{2}$ so the two instances are hard to tell apart. In particular, if $|\widetilde{v}^\pi - v^\pi| \geq a + \tilde{a}$, then at least one of the following holdsone of the following must hold:

$$\mathbb{P}_p\Big(|v^\pi - \widehat{v}^\pi(\mathscr{D})| \geq a\Big) \geq \frac{1}{6} \text{or } \mathbb{P}_{\tilde{p}}\Big(|\tilde{v}^\pi - \widehat{v}^\pi(\mathscr{D})| \geq \tilde{a}\Big) \geq \frac{1}{6}$$

- **Optimizing the perturbation direction:** The value gap between the two instances is

$$\widetilde{v}^\pi - v^\pi \approx \sum_{h=0}^{H} \xi_0^\top (\mathbf{P}^\pi)^h (\widetilde{\mathbf{P}}^\pi - \mathbf{P}^\pi) Q_{h+1}^\pi \gtrsim \sum_{h=0}^{H-1} (H-h) \mathbb{E}^\pi[\phi(s_h, a_h)]^\top \mathbf{x}$$

- Maximizing the RHS above with the constraint $\sqrt{\mathbf{x}^\top \Sigma \mathbf{x}} \lesssim N^{-1/2}$, we obtain $\mathbf{x}^*$ and the corresponding $\tilde{p}$. QED

Does regression also work for online exploration in RL?

Learning to Control On-The-Fly

- Prior work assumes a **generative model** (guaranteed exploration) or **batch data** (no exploration)
- In practice, we have to *learn on-the-fly without any simulator.*
- **Episodic RL:**
  - H-horizon stochastic control problem, starting at a fixed state s_0
  - A learning algorithm learns to control adaptively by repeatedly acting in the real world
  - Impossible to visit all representative states frequently
  - This is an adaptive control problem

Episodic Reinforcement Learning

- **Regret of a learning algorithm $\mathscr{K}$**

$$\textbf{Regret}_{\mathscr{K}}(T) = \mathbb{E}_{\mathscr{K}}\left[\sum_{n=1}^{N}\left(V^*(s_0) - \sum_{h=1}^{H} r\left(s_{n,h}, a_{n,h}\right)\right)\right],$$

where T= NH, and the sample state-action path $\{s_{n,h}, a_{n,h}\}$ is generated on-the-fly by the learning algorithm

- **Theoretical challenges:**
  - Long-term effect of a single wrong decision
  - Data dependency: Almost all the transition samples are dependent
  - Exploration-exploitation tradeoff

Lots of pioneering works and milestones:

- (Kaelbling 1995), (Strens 2000), (Auer &Otner 2007), (Abbasi-Yadkori & Szepesvári 2011), (Osband & Van Roy 2014), (Zheng and Van Roy 2013), (Jin et al 2018), (Russo 2019) and many others

Feature space embedding of transition kernel

- Suppose we are given state-action feature maps (or kernels)

$$state, action \mapsto [\phi_1(state, action), \ldots, \phi_d(state, action)] \in \mathbb{R}^N$$

$$state \mapsto [\psi_1(state), \ldots, \psi_{d'}(state)] \in \mathbb{R}^{d'}$$

- Assume that the unknown transition kernel can be fully embedded in the feature space, i.e., there exists a transition core matrix M* such that

$$\phi(s,a)^\top M^* = \mathbf{E}[\psi(s')^\top \mid s, a] \, .$$

- Also assume that ψ is sufficient to express any value function

- Let's borrow ideas from linear bandit (Dani et al 08, Chu et al 11, many others)

MatrixRL algorithm

- **Model estimation via matrix ridge regression** (aka conditional mean embedding matrix)

$$M_n = \textbf{argmin}_M \sum_{t=1}^{nK} \|\phi(s_t, a_t)^\top M - \psi(s_t')\|^2 + \lambda \|M\|_F^2$$

- **Construct a matrix confidence ball**
- $$B_n = \left\{ M \in \mathbb{R}^{d\times d'} : \quad \|(\sum_{t=1}^{nK} \phi(s_t, a_t)\phi(s_t, a_t)^\top)^{1/2}(M - M_n)\|_F \le \sqrt{\beta_n} \right\}$$
- **Find optimistic Q-function estimate**
- $$Q_{n,h}(s,a) = r(s,a) + \max_{M\in B_n} \phi(s,a)^\top M w_{n,h+1}, \quad Q_{n,H} = 0$$

where $w_{n,h+1}$ is the low-dim representation of value estimate $V_{n,h}(s) = \max_a Q_{n,h}(s,a)$

- **In the new episode, always choose actions greedily by argmax$_a Q_{n,h}(s,a)$**
- The optimistic Q encourage exploration: (s,a) with higher uncertainty gets tried more often

Another view of MatrixRL

- **Feature maps ϕ, ψ define the families for approximating Q and V functions**
- MatrixRL has a closed-form update, which is an optimistic Q-leaning update

$$\hat{Q}_h(s,a) \leftarrow r_{s,a} + \phi(s,a)^\top \hat{w} + poly(H)\sqrt{\phi(s,a)^\top \Sigma^{-1} \phi(s,a)}$$

$$\boxed{\approx \textbf{argmin}_w \sum_{(s,a,s') \in Experiences} (\phi(s,a)^\top w - \hat{V}_{h+1}(s'))^2}$$

optimism bonus

- The regression step hides the matrix regression
- Reduce the T-step regret of exploration in RL:

$$\sqrt{poly(H)SAT} \quad \text{reduces to} \quad H^2 poly(d)\sqrt{T}$$

Regret Analysis

- **Theorem:** Under the embedding assumption and regularity assumptions, the T-time-step regret of MatrixRL satisfies with high probability

$$\text{Regret}(T) \leq C \cdot dH^2 \cdot \sqrt{T}$$

- The method can be *kernelized to work with any RKHS:*

$$\textit{Regret}(T) \leq O\left(\|P\|_{\mathbf{H}_\phi \times \mathbf{H}_\psi} \cdot \log(T) \cdot \tilde{d} \cdot H^2 \cdot \sqrt{T} \right)$$

- *where* $\tilde{d}$ *is an effective dimension*

- First polynomial regret bound for RL in nonparametric kernel space

(RL in Feature Space: Matrix Bandit, Kernels, and Regret Bounds, Preprint, 2019)

How to conduct regression efficiently in end-to-end training of RL agents?

Doing the right regression is nontrivial

- In MatrixRL, the algorithm is essentially training a model predictor

$$\hat{f} : \psi(s, a) \rightarrow \hat{\mathbb{E}}[\phi(s')]$$

- To make this work in an actual RL task, one needs to specify the regression target $\phi(s')$

- A common example is the raw next state (eg. raw-pixel images)

Challenges of pixel-to-pixel training:

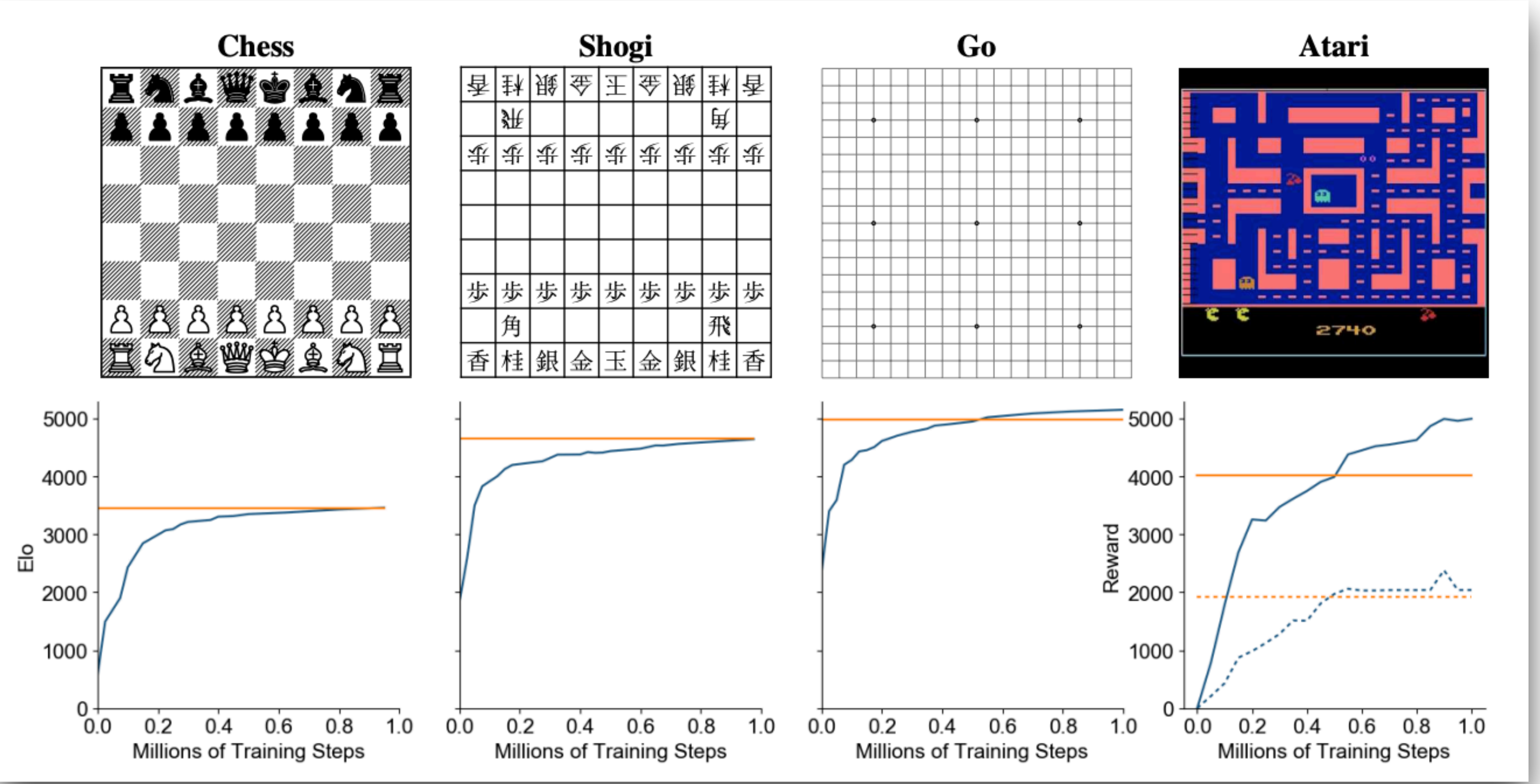
- Much of the predicted quantities are not relevant to solving the game

- Scaling/transforming the target is necessary and requires case-by-case tuning

- Computation overhead and poor generalizability

A motivating example: *MuZero*

A single algorithm generalizes to 60 games and beats the best player of each



End-to-end training; no prior knowledge of game rules; plan & explore with a learned model

Key idea: only try to predict quantities central to the game, e.g., value and policies

(figure from MuZero paper, DeepMind 2019)

More general model-based RL

- Suppose we have a general class of transition models $\mathscr{P} = \{P_\theta \,|\, \theta \in \Theta\}$
- A general framework for optimistic model-based RL:

    1. Given past data $\mathscr{D}$, construct a confidence set : $B \leftarrow \{\theta \,|\, L(\theta; \mathscr{D}) \leq \beta\}$
    2. Optimistic planning with a learned model: $\sup_{\theta \in B} V_\theta(s_0)$

Some questions:

- Is it necessary to recover the full transition model?
- Can we do model predictive control without predicting the actual state?
- Can we only use value functions for self-training?
- How to construct the loss function L?

  Short answer: yes

Exploration with Value-Targeted Regression (VTR)

- Let $\hat{V}$ be current value function at the beginning of a new episode.

1. Whenever observing a new sample (s, a, r', s'), update data buffer
   $D \leftarrow D \cup \{(x(\cdot), y)\}$ where $x(\theta) = \mathbb{E}_\theta[\hat{V}(s') \,|\, s, a], y = \hat{V}(s')$

2. Value-targeted nonlinear regression $\hat{\theta} = \text{argmin}_\theta \sum_{(x,y)\in\mathcal{D}} (x(\theta) - y)^2$

3. Planning using an optimistic learned model
   $\theta_{opt} \leftarrow \text{argmax}_{\theta\in\mathcal{B}} V_\theta(s_0)$, where $\mathcal{B} = \left\{ \theta \,\middle|\, \sum_{(x,y)\in\mathcal{D}} (x(\theta) - x(\hat{\theta}))^2 \leq \beta \right\}$
   $\hat{\pi} \leftarrow \text{argmax}_\pi V^\pi_{\theta_{opt}}(s_0)$, $\hat{V} \leftarrow V^{\hat{\pi}}_{\theta_{opt}}$,

- Implement $\hat{\pi}$ as the policy in the next run

- The target value function $\hat{V}$ keeps changing as the agent learns

(Model-based RL with Value Targeted Regression. with Szepesvari, Yang et al. Preprint, 2020)

Regret analysis of VTR

Theorem: By choosing confidence levels $\{\beta_k\}$ appropriately, the VTR algorithm's regret satisfies with probability $1 - \delta$ that

$$R_K = \sum_{k=1}^{K} \left(V^*(s_0^k) - V^{\hat{\pi}_k}(s_0^k) \right) \leq \tilde{O}(\sqrt{dim_{\mathscr{E}}(\mathscr{F}, 1/KH) \log \mathscr{N}(\mathscr{F}, 1/KH^2, \|\cdot\|_{1,\infty}) KH^3})$$

where $dim_{\mathscr{E}}(\mathscr{F}, 1/KH)$ is the Eluder dimension (Russo & Van Roy 2013) of the function class

$$\mathscr{F} = \{f : \mathscr{S} \times \mathscr{A} \times \mathbb{R}^{\mathscr{S}} : \exists \theta \in \Theta s.t. f(s, a, v) = \int p_\theta(s' \mid s, a) f(s') d(s, a)$$

and $\mathscr{N}(\mathscr{F}, \alpha, \|\cdot\|_{1,\infty})$ denotes the covering number of $\mathscr{F}$ at a the scale α.

- First frequentist regret bound for model-based RL with a general model class
- Matches the bayesian regret using posterior setting (Osband & Van Roy 2014). In the special case of linear-factor model, matches the results of (Yang & Wang, 2019) (Jin et al, 2019)

Value-targeted regression is efficient for exploration in RL

Summary

- When “good” state-action features are given, the minimax-optimal sample complexity of MDP (with a generative model) reduces by

$$\Theta\left(\frac{|SA|}{(1-\gamma)^3\epsilon^2}\right) \to \Theta\left(C \cdot \frac{|d_S d_A|}{\epsilon^2}\right)$$

- Regression-based plug-in estimator is near-optimal for batch-data policy evaluation

$$\inf_{\hat{v}^\pi} \sup_{M,\bar{\pi}} |\,\widehat{v}^\pi - v^\pi| \asymp H^2 \sqrt{\frac{1+\chi^2_{\mathcal{Q}}(\mu^\pi, \bar{\mu})}{N}} + o(1/\sqrt{N})$$

- Value-targeted regression is efficient to model-based RL

$$R_K \leq \tilde{O}(\sqrt{dim_{\mathscr{E}}(\mathscr{F}, 1/KH) \log \mathcal{N}(\mathscr{F}, 1/KH^2, \|\cdot\|_{1,\infty}) KH^3})$$

Good news: Regression works!

Thank you!