“The essence of training is to allow error without consequence.”

Orson Scott Card, Ender’s Game

CMPUT 628
Deep RL

Marlos C. Machado

https://robert-simons-pevb.squarespace.com/enders-game-ships

Class 22/ 25

Reminders & Notes

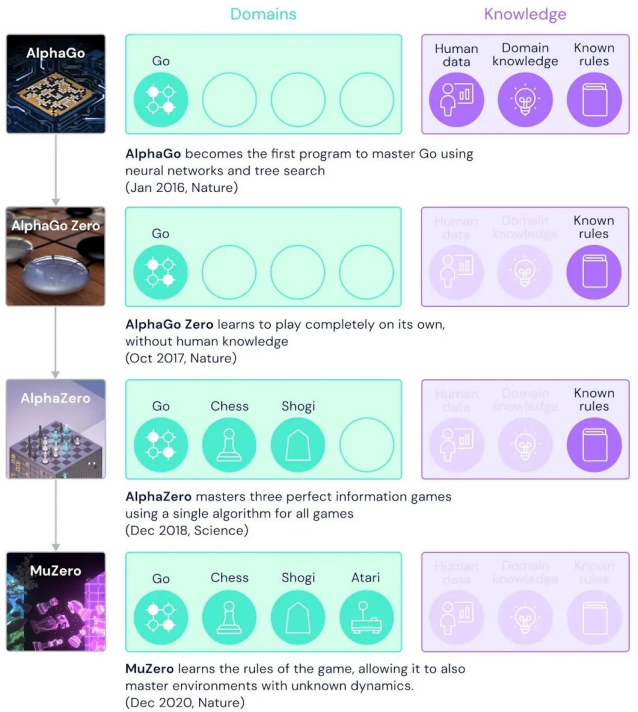
- The Midterm is next class, on Tuesday
- Seminars will start after that

Please, interrupt me at any time!

https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

One of the most famous results in deep RL is “model-based”

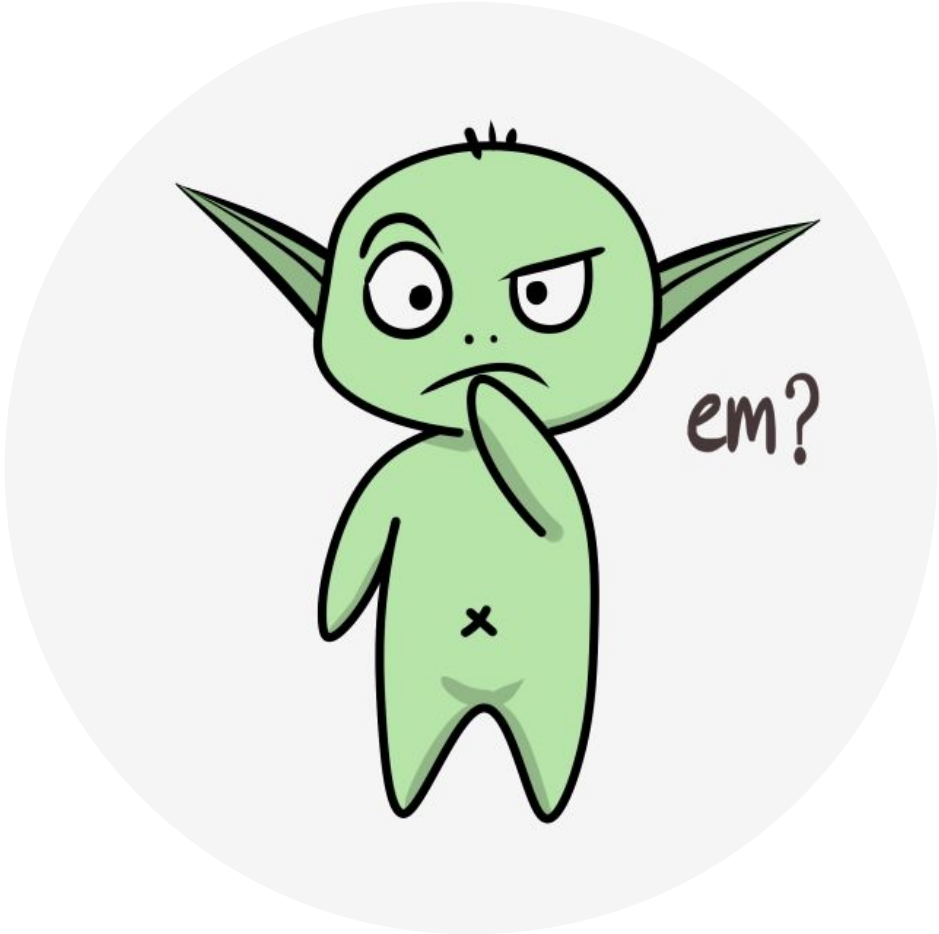
AlphaGo: *“China’s Sputnik Moment”*



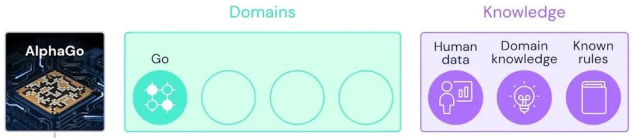
https://deepmind.google/discover/blog/muzero-mastering-go-chess-shogi-and-atari-without-rules/

And DeepMind sold AlphaGo amazingly well

https://www.youtube.com/watch?v=WXuK6gekU1Y

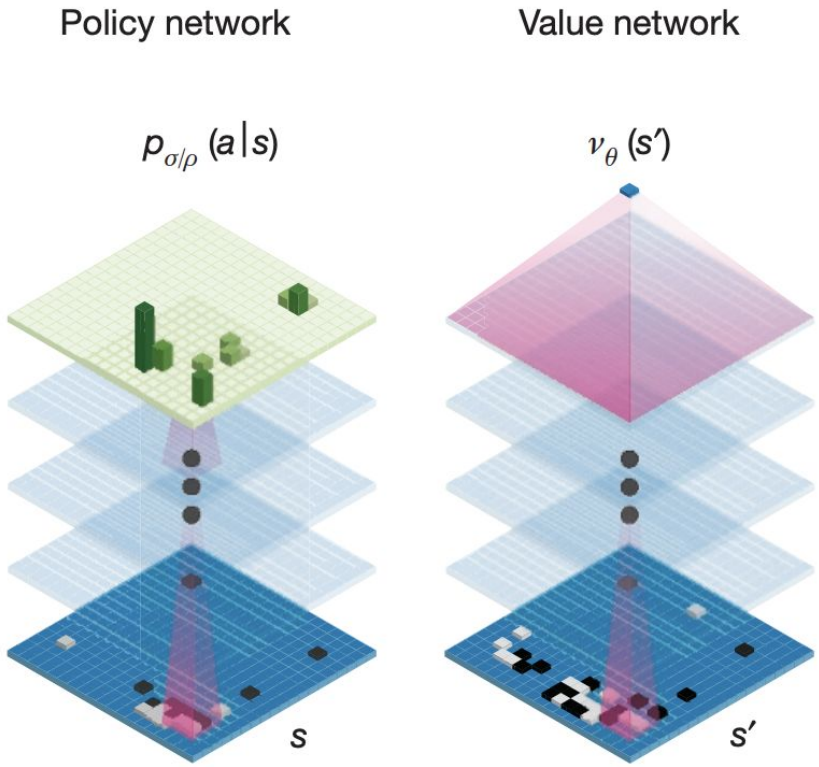


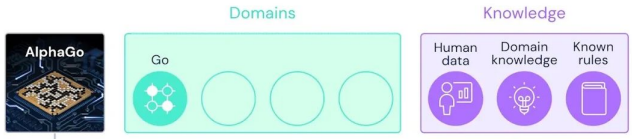
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht



AlphaGo [Silver et al., 2016]

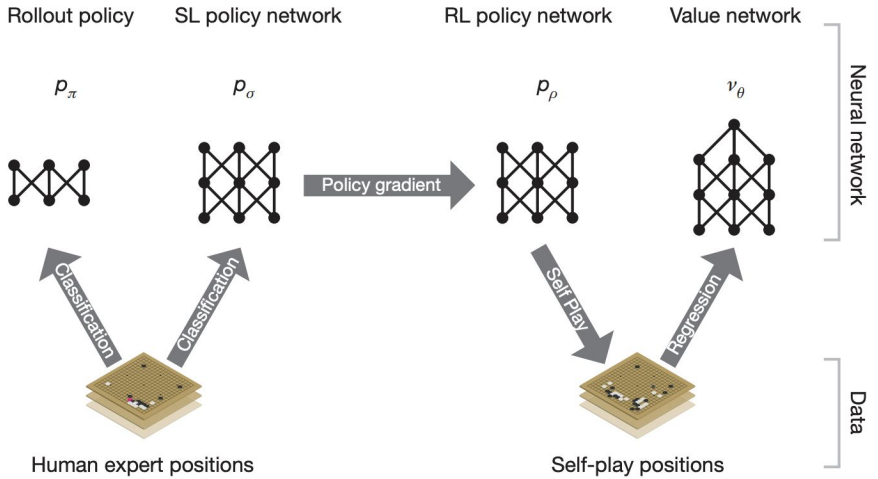
- MCTS. Value estimation (bootstrapping) allows us to reduce depth, and a policy allows us to reduce the breadth of the search space
- *Value networks* evaluate board positions and *policy networks* select moves
- Combine policy and value network in an MCTS algorithm

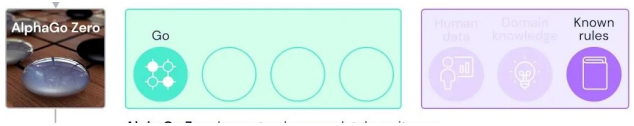




AlphaGo [Silver et al., 2016]

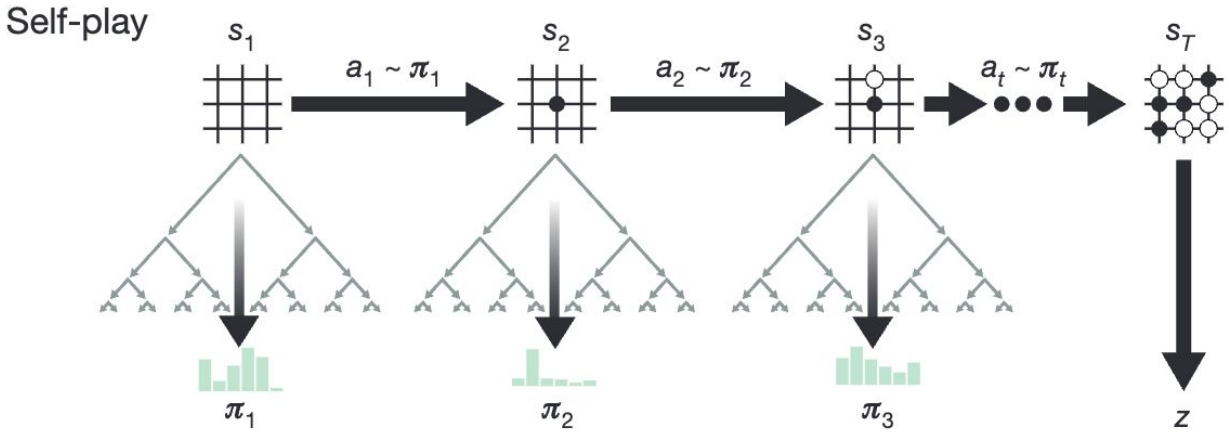
1. Train supervised learning policy network from expert data
2. Train a fast policy that can sample actions during rollouts
3. Train an RL policy network that improves 1 through self-play
   a. Moves away from predicting experts moves toward winning games
4. Train value network to predict the winner of games played by the RL policy network against itself
5. Combine policy and value network in an MCTS algorithm

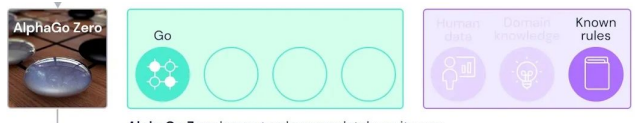




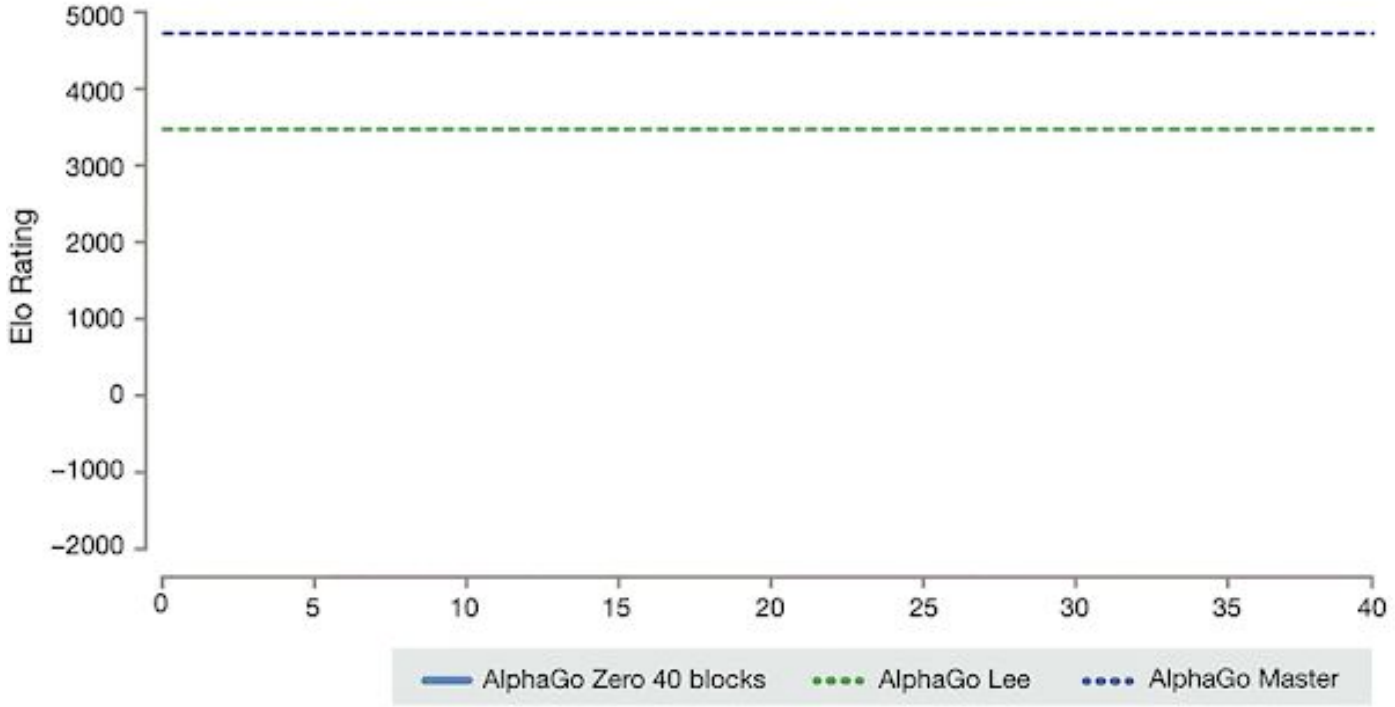
AlphaGo Zero [Silver et al., 2017]

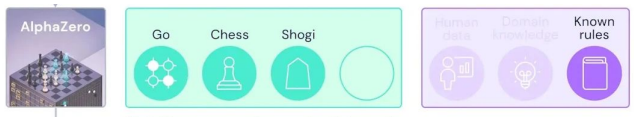
- AlphaGo but with no supervisor, no human data, just self-play
- No domain knowledge in the agent input
- Single neural network (ResNet with batch normalization) with two heads
  - “the dual objective regularizes the network to a common representation that supports multiple use cases”
- Simpler tree search





AlphaGo Zero [Silver et al., 2017]

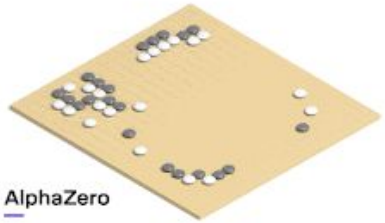


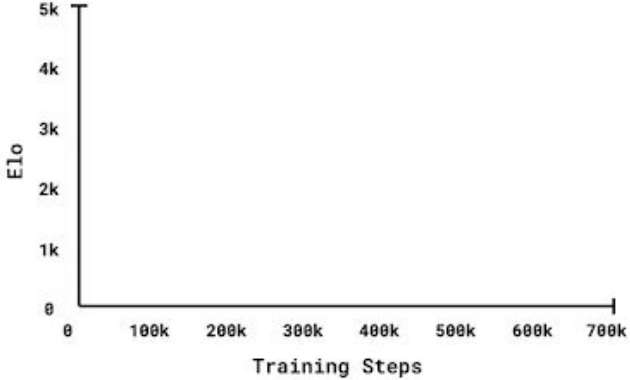


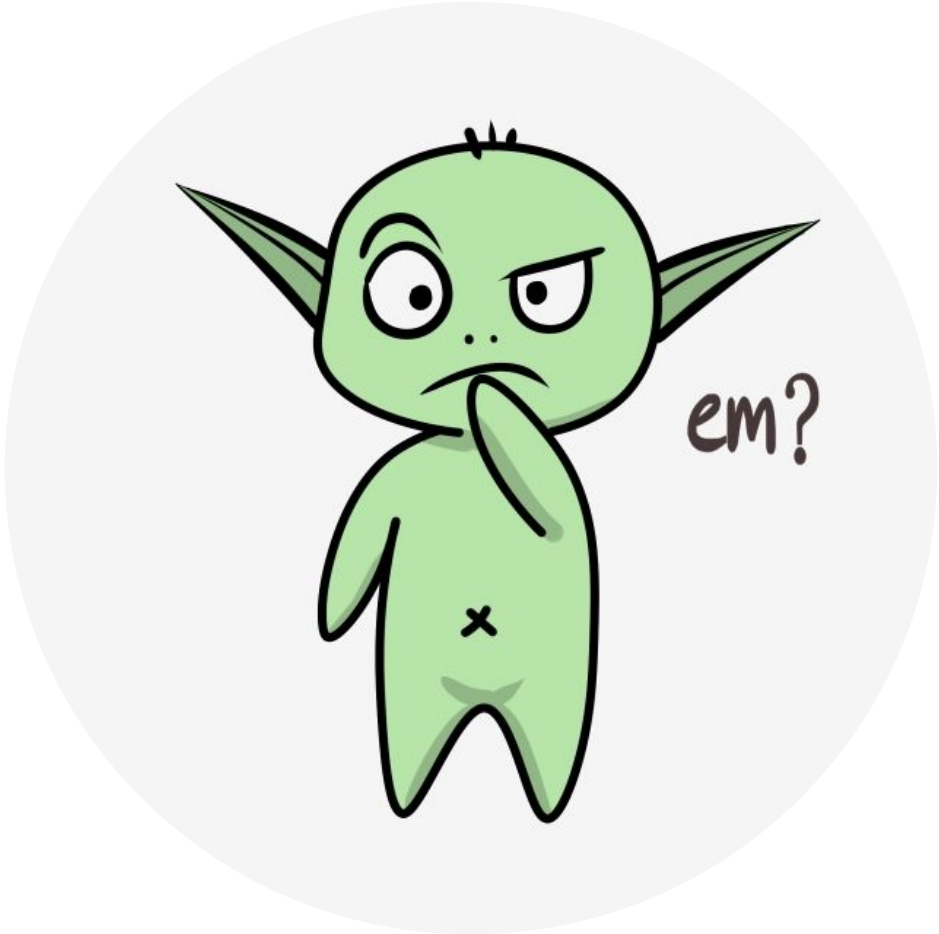
AlphaZero [Silver et al., 2018]

- Pretty much AlphaGo Zero, but with some adjustments to include more games
  - AlphaZero estimates/optimizes the expected outcome, not winning probability
  - Updates neural network continually instead of doing it in iterations
    (iterations were chosen to select the best player in AlphaGo Zero)
  - Does not assume symmetry

But AlphaGo, AlphaGo Zero, and AlphaZero still rely on a given model for the MCTS!



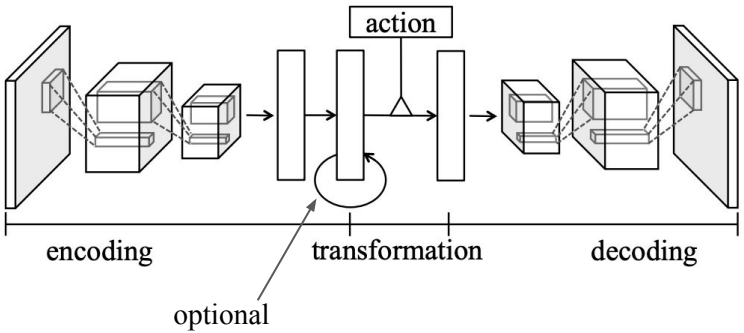


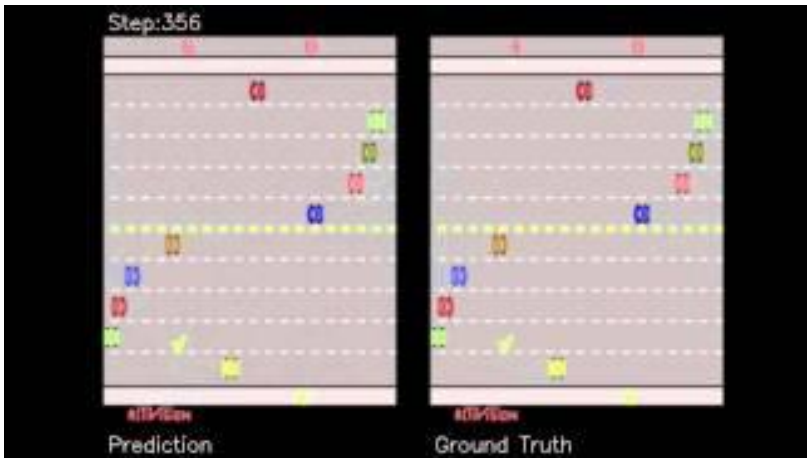


https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Model-Based Deep Reinforcement Learning

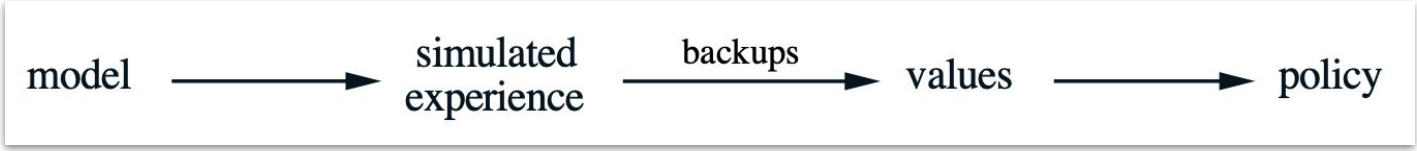
- Not much long after DQN was introduced, in 2013, we had neural networks being used in PG methods (e.g., DDPG, TRPO) and model-based RL
- Oh et al. (2015), for example, already have impressive mode learning results (*but no planning results*)





What we want to use a model for?

- We can use it to do exploration, as an auxiliary task, and so on
  Here, let's focus on planning; thus: to ultimately improve the policy



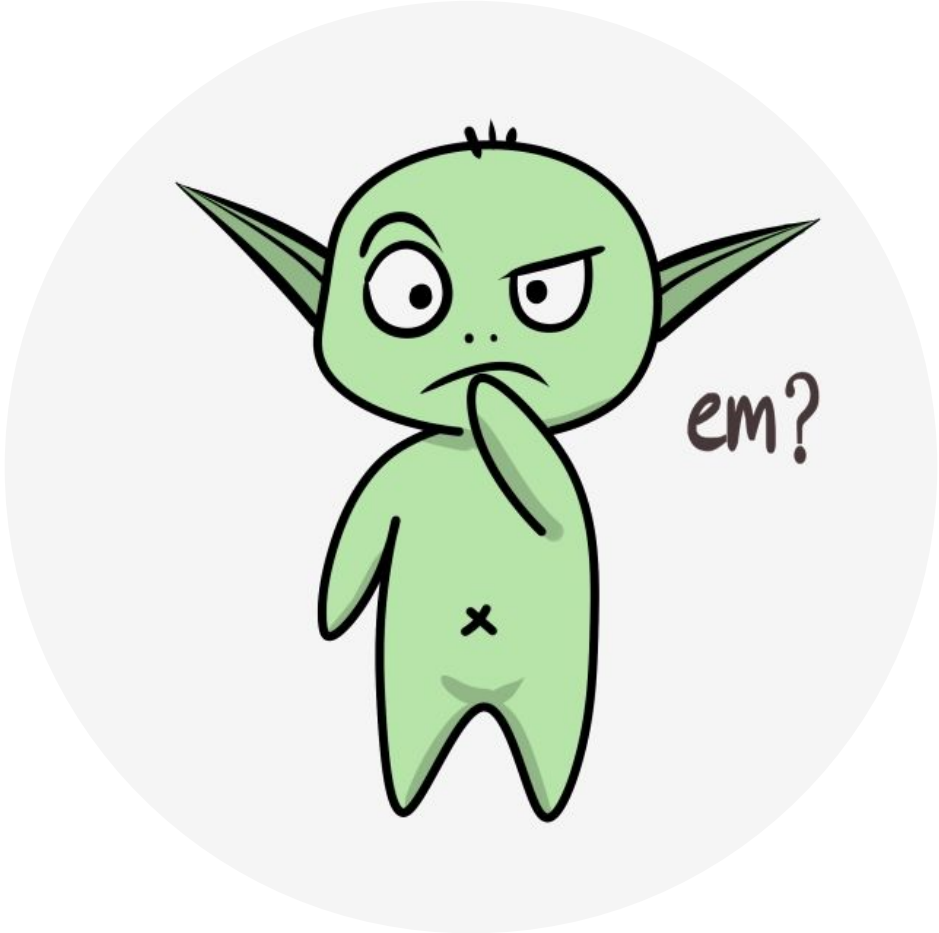
- There are different types of planning:
  - *Backward planning:* We generate samples so that we can use additional samples to improve our value estimates so we get a better policy
  - *Forward planning:* We look ahead thinking about many of the decisions we can make, sort of doing search with a learned model, so we have a better idea of which action to take next

Backward Planning

- Given a state, *S*, imagine the consequence (i.e., next state and reward) of taking a specific action, *A*
- Dyna-Q (Sutton, 1990), for example, does exactly that (in the *tabular case*) Dyna-Q learns a model to generate additional samples so we can do Q-learning (or Q-planning) updates
- Experience replay buffers already allow us to replay transitions, though, making it harder for us to observe the benefits of model-based reinforcement learning
  - *Generalization beyond seen trajectories / Avoiding policy staleness:* We can go beyond seen transitions (counterfactuals?)
  - *Learning in Latent Space:* Generalization across states
  - More?

Forward Planning

- Given a state *S*, simulate multiple possible action sequences from *S* to estimate future outcomes and select the best action accordingly
- MCTS [Metropolis & Ulam, 1949] and MPC [Richalet et.al, 1976; Cutler & Ramaker, 1980] are some of the techniques we can use on top of a *learned* model
- Forward planning is offering us something fundamentally different from experience replay buffers
  - *Efficient exploration:* One can look at multiple futures and backtrack from choices
  - *More robust to model errors:* Replans dynamically, which is quite useful with inaccurate models
  - *Better value estimates:* Bootstrapping has a smaller role to play when looking multiple steps ahead



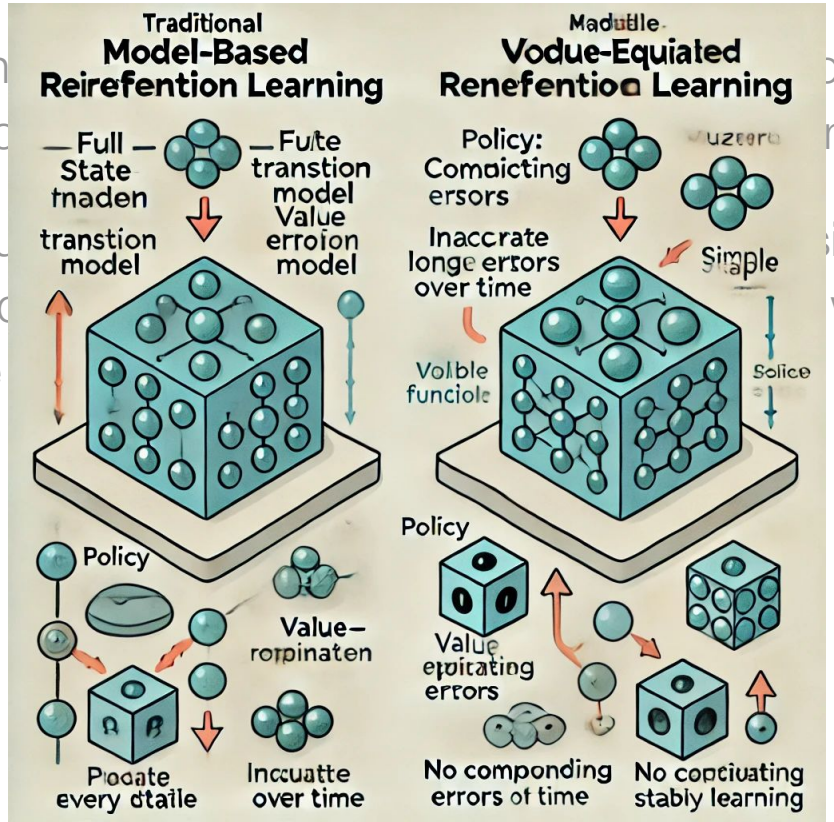
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Why many MBRL methods struggle?

- Many MBRL methods focus on learning a full transition model but suffer from compounding errors and unnecessary complexity (e.g., SimPLe [Kaiser et al., 2020])
- Arguably, we should be using methods that explicitly consider decision-making by having value-equivalent modeling, that is, learning only what's useful for good policies and value estimation

ChatGPT: Why many MBRL methods struggle?

- Many MBRL meth… …odel but suffer from compounding erro… …mPLe [Kaiser et al., 2020])
- Arguably, we shou… …sider decision-making by having value-ec… …what's useful for good policies and value …



Left Side (Traditional MBRL): Learns full transitions but suffers from compounding errors.

Right Side (Value-Equivalent MBRL): Learns models optimized for decision-making.

The Value Equivalence Principle [Grimm et al., 2020]

- Models don't have to be accurate, they have to be useful
- "two models are value equivalent with respect to a set of functions and a set of policies if they yield the same updates under corresponding Bellman operators"

Definition 1 (Value equivalence). *Let $\Pi \subseteq \mathbb{\Pi}$ be a set of policies and let $\mathcal{V} \subseteq \mathbb{V}$ be a set of functions. We say that models m and $\tilde{m}$ are* value equivalent *with respect to Π and $\mathcal{V}$ if and only if*

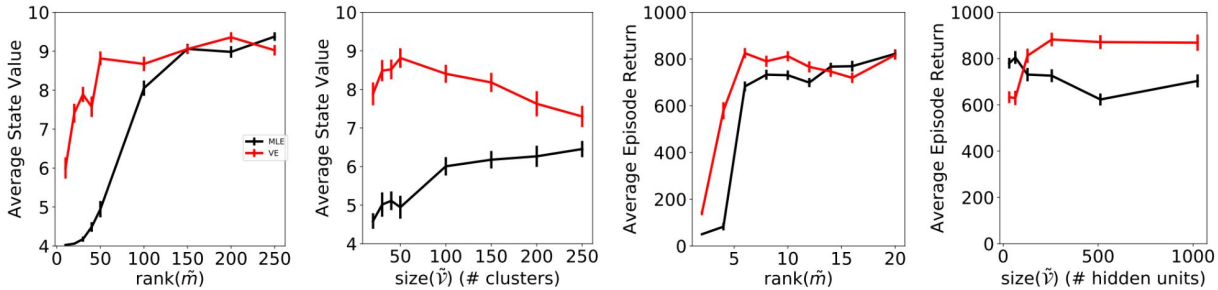
$$\mathcal{T}_\pi v = \tilde{\mathcal{T}}_\pi v \text{ for all } \pi \in \Pi \text{ and all } v \in \mathcal{V},$$

where $\mathcal{T}_\pi$ and $\tilde{\mathcal{T}}_\pi$ are the Bellman operators induced by m and $\tilde{m}$, respectively.

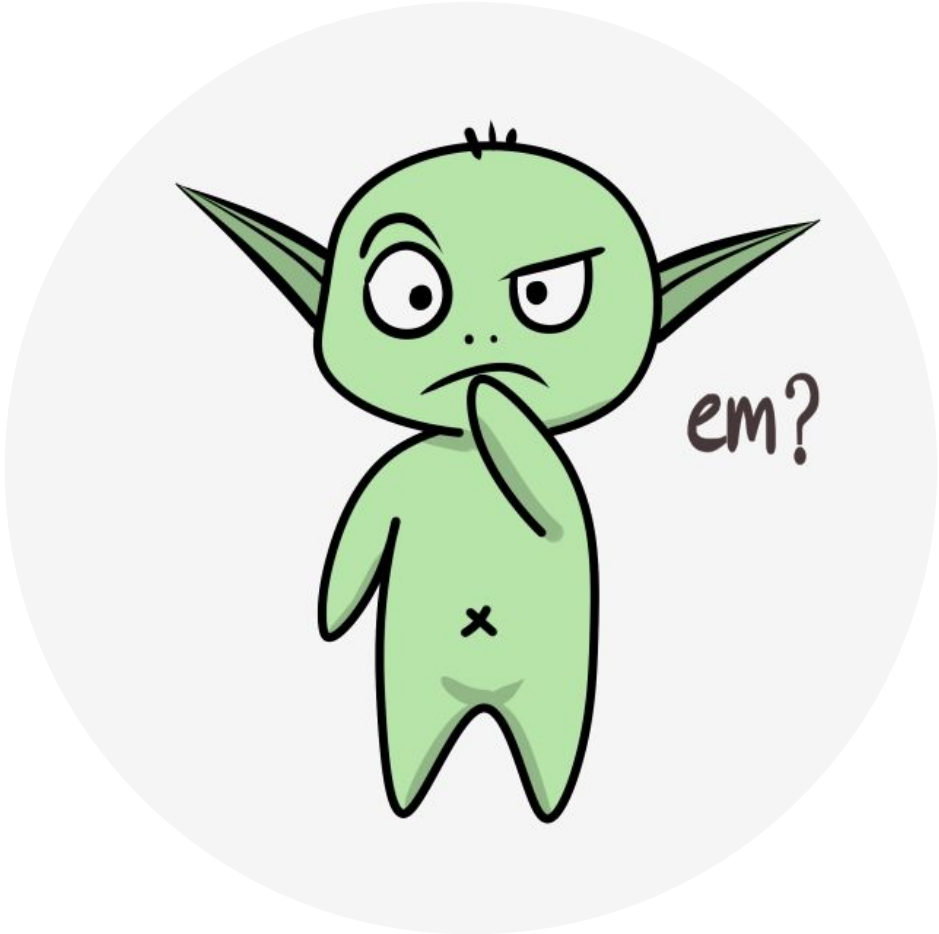
The Value Equivalence Principle [Grimm et al., 2020]

Proposition 3. *The maximum-likelihood estimate of p^* in $\mathcal{P}$ may not belong to a $\mathcal{P}(\Pi, \mathcal{V}) \neq \emptyset$.*

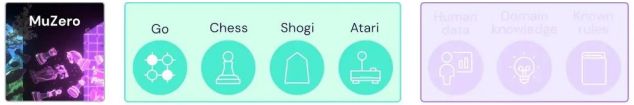
- "Proposition 3 states that (...) even when it is possible to perfectly handle the policies in Π and the values in V, the model that achieves the smallest MLE loss will do so only approximately."



(a) Catch (fixed $\mathcal{V}$) (b) Catch (fixed $\tilde{m}$) (c) Cart-pole (fixed $\mathcal{V}$) (d) Cart-pole (fixed $\tilde{m}$)



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht



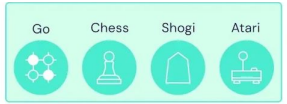
MuZero [Schrittwieser et al., 2020]

- MuZero models three elements of the agent that are critical to planning, ***avoiding full transition learning***
  - Value, policy, and reward

- MuZero is defined for ***deterministic*** dynamics
  - It uses a recurrent notion state to keep track of where it is, but with no semantics about the state itself
- Similar to AlphaZero, it has a network with many heads, represented by the function *f*

*"MuZero uses the representation function, **h**, to map from the observation to an embedding used by the neural net. Using the dynamics function, **g**, and the prediction function, **f**, MuZero can then consider possible future sequences of actions and choose the best action."*

https://deepmind.google/discover/blog/muzero-mastering-go-chess-shogi-and-atari-without-rules/

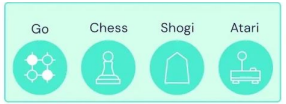


MuZero [Schrittwieser et al., 2020]

- Similar to AlphaZero, MuZero uses MCTS; but here it does the search within its internal dynamics
- But it allows for discounting and it uses *n*-step returns

"MuZero uses the experience it collects when interacting with the environment to train its neural network. This experience includes both observations and rewards from the environment, as well as the results of searches performed when deciding on the best action."

https://deepmind.google/discover/blog/muzero-mastering-go-chess-shogi-and-atari-without-rules/



MuZero [Schrittwieser et al., 2020]

- It is trained to minimize the three obj. at the same time, with ℓ_2 regularization

$$l_t(\theta) = \sum_{k=0}^{K} l^{\mathrm{p}}(\pi_{t+k}, p_t^k) + \sum_{k=0}^{K} l^{\mathrm{v}}(z_{t+k}, v_t^k) + \sum_{k=1}^{K} l^{\mathrm{r}}(u_{t+k}, r_t^k) + c\|\theta\|^2$$

"During training, the model is unrolled alongside the collected experience, at each step predicting the previously saved information: the value function v predicts the sum of observed rewards, u, the policy estimate, p, predicts the previous search outcome, π, the reward estimate r predicts the last observed reward, u."

https://deepmind.google/discover/blog/muzero-mastering-go-chess-shogi-and-atari-without-rules/

MuZero works really, really well [Schrittwieser et al., 2020]

- "We trained MuZero for K = 5 hypothetical steps. Training proceeded for one million mini-batches of size 2,048 in board games and of size 1,024 in Atari. During both training and evaluation, MuZero used 800 simulations for each search in board games and 50 simulations for each search in Atari."

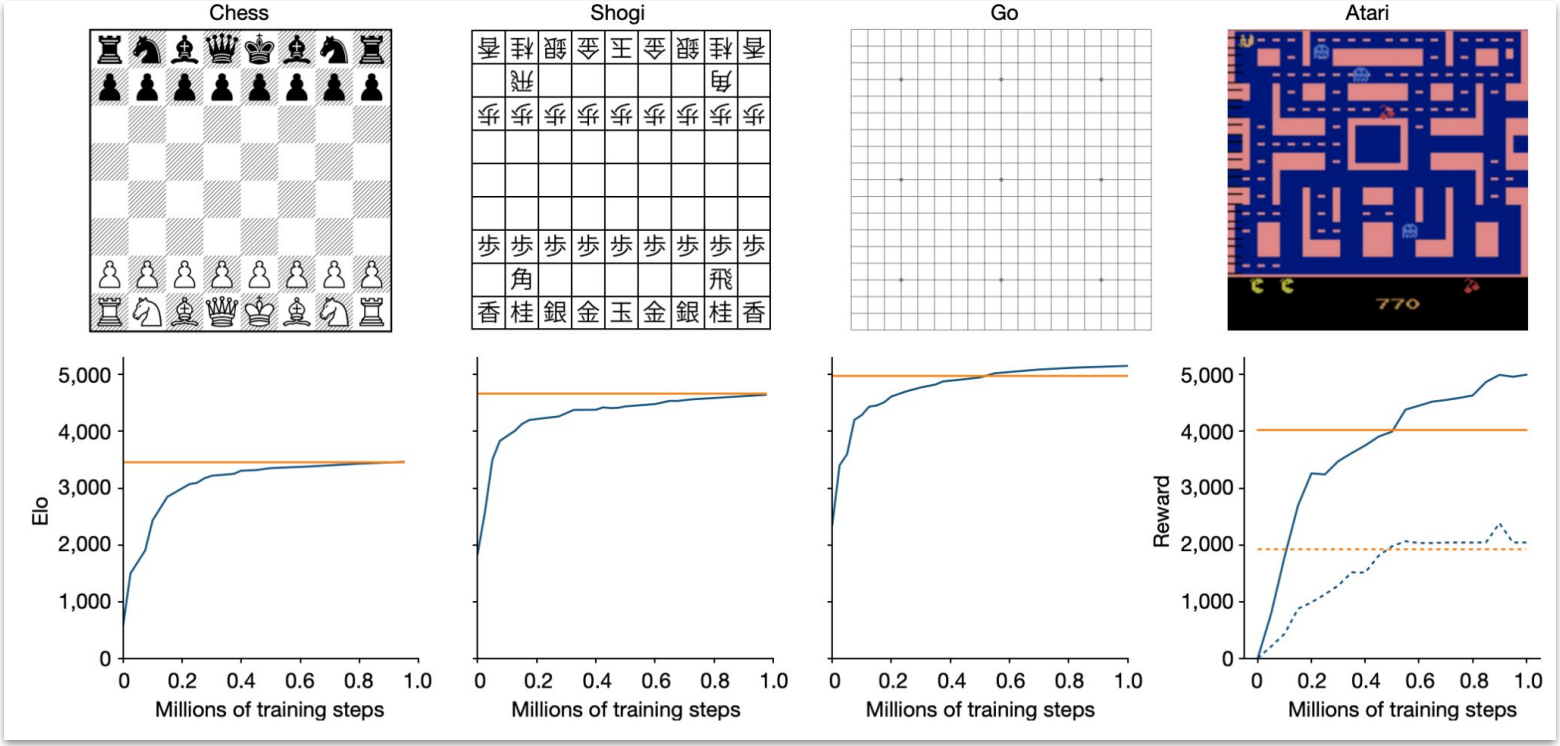
Table 1 | Comparison of MuZero against previous agents in Atari

Agent	Median (%)	Mean (%)	Environment frames	Training time	Training steps
Ape-X[20]	434.1	1,695.6	22.8 billion	5 days	8.64 million
R2D2[19]	1,920.6	4,024.9	37.5 billion	5 days	2.16 million
MuZero	**2,041.1**	**4,999.2**	20.0 billion	12 hours	1 million
IMPALA[18]	191.8	957.6	200 million	–	–
Rainbow[36]	231.1	–	200 million	10 days	–
UNREAL[a 42]	250[a]	880[a]	250 million	–	–
LASER[37]	431	–	200 million	–	–
MuZero Reanalyze	**731.1**	**2,168.9**	200 million	12 hours	1 million

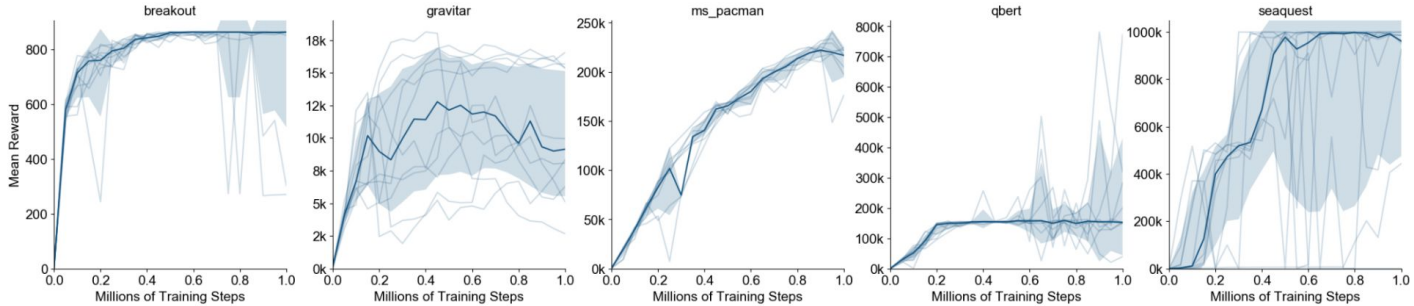
We compare separately against agents trained in large (top) and small (bottom) data settings; all agents other than MuZero used model-free RL techniques. Mean and median scores are given, compared with human testers. The best results are highlighted in bold. MuZero shows state-of-the-art performance in both settings. [a]Hyperparameters were tuned per game.

"reanalyses old trajectories by re-running the MCTS using the latest network parameters to provide fresh targets"

MuZero works really, really well [Schrittwieser et al., 2020]



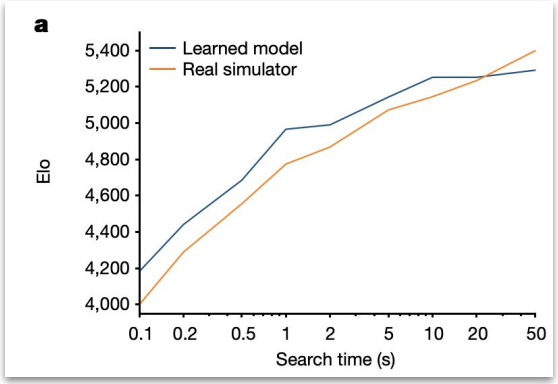
But it can have quite a lot of variance [Schrittwieser et al., 2020]

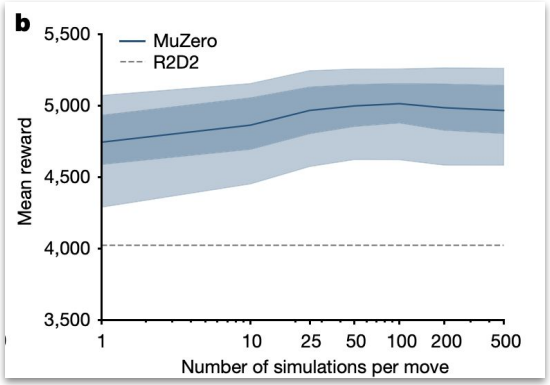


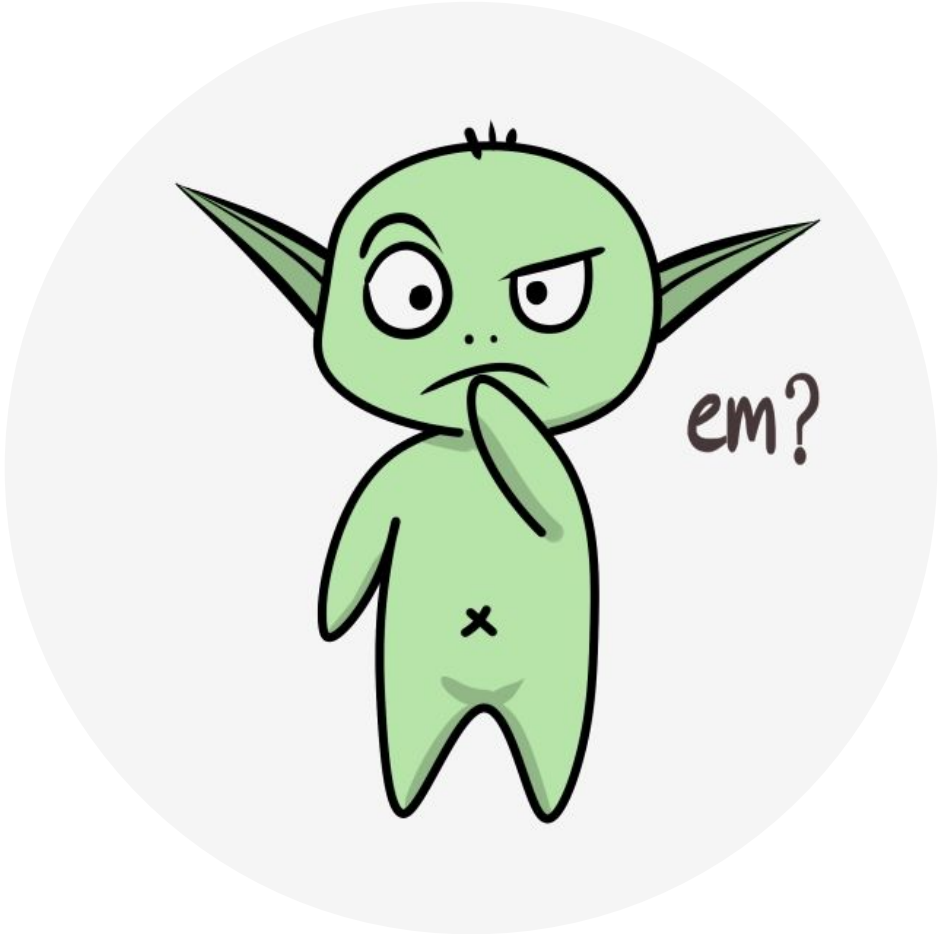
	breakout	gravitar	ms pacman	qbert	seaquest
μ	747.2	10,338.8	210,060.0	237,393.8	806,233.8
σ	230.2	4,790.4	15,520.8	193,266.1	330,129.0

The impact of MuZero's search in Atari games [Schrittwieser et al., 2020]

"We compared MCTS with different numbers of simulations, using the fully trained MuZero. The improvements due to planning are much less marked than in Go, perhaps because of greater model inaccuracy; performance improved slightly with search time, but plateaued at around 100 simulations. Even with a single simulation—that is, when selecting moves solely according to the policy network—MuZero performed well, suggesting that, by the end of training, the raw policy has learned to internalize the benefits of search ."



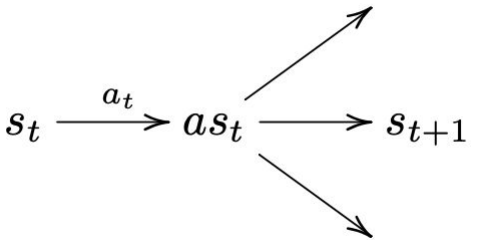




https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Stochastic MuZero [Antonoglou et al., 2022]

- "An afterstate as_t is the hypothetical state of the environment after an action is applied but before the environment has transitioned to a true state"
- "By using afterstates we can separate the effect of applying an action to the environment and of the chance transition given an action"

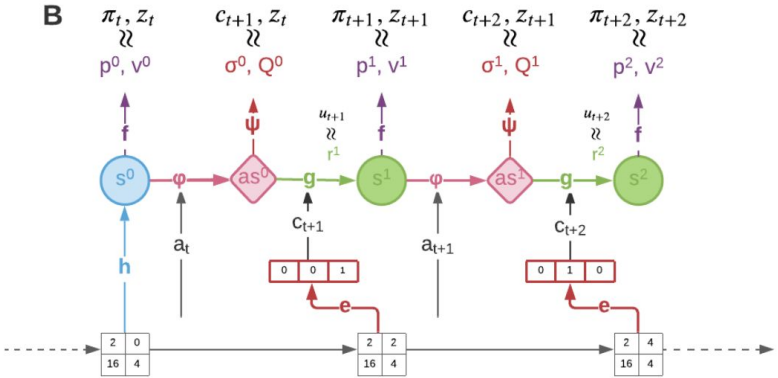


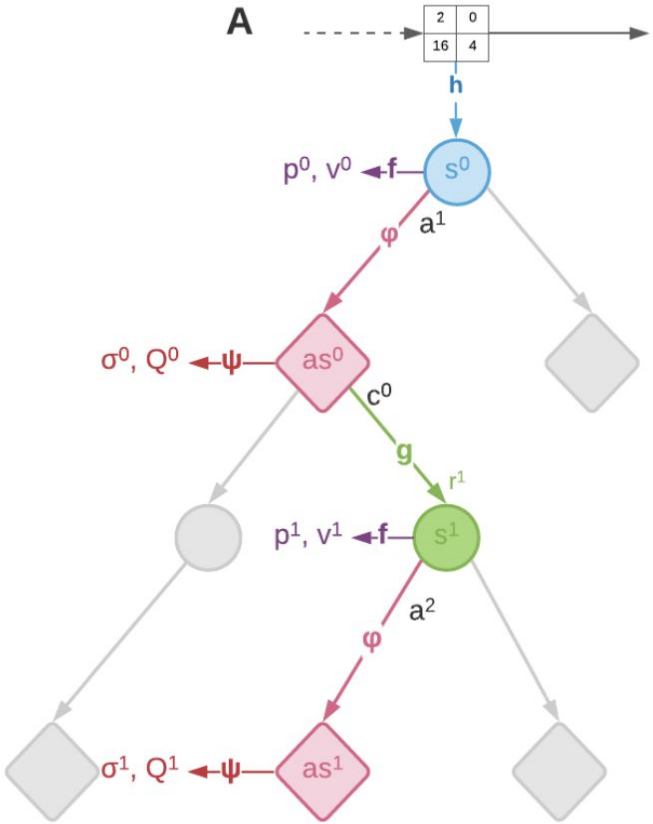
- By defining afterstates, as_t, and chance outcomes, c_t, we can model a chance transition using a deterministic model $s_{t+1}, r_{t+1} = M(as_t, c_t)$ and a distribution $Pr(s_{t+1} \mid as_t) = Pr(c_t \mid as_t)$. The task of learning a stochastic model is then reduced to the problem of learning afterstates, as, and chance outcomes, c.

Stochastic MuZero [Antonoglou et al., 2022]

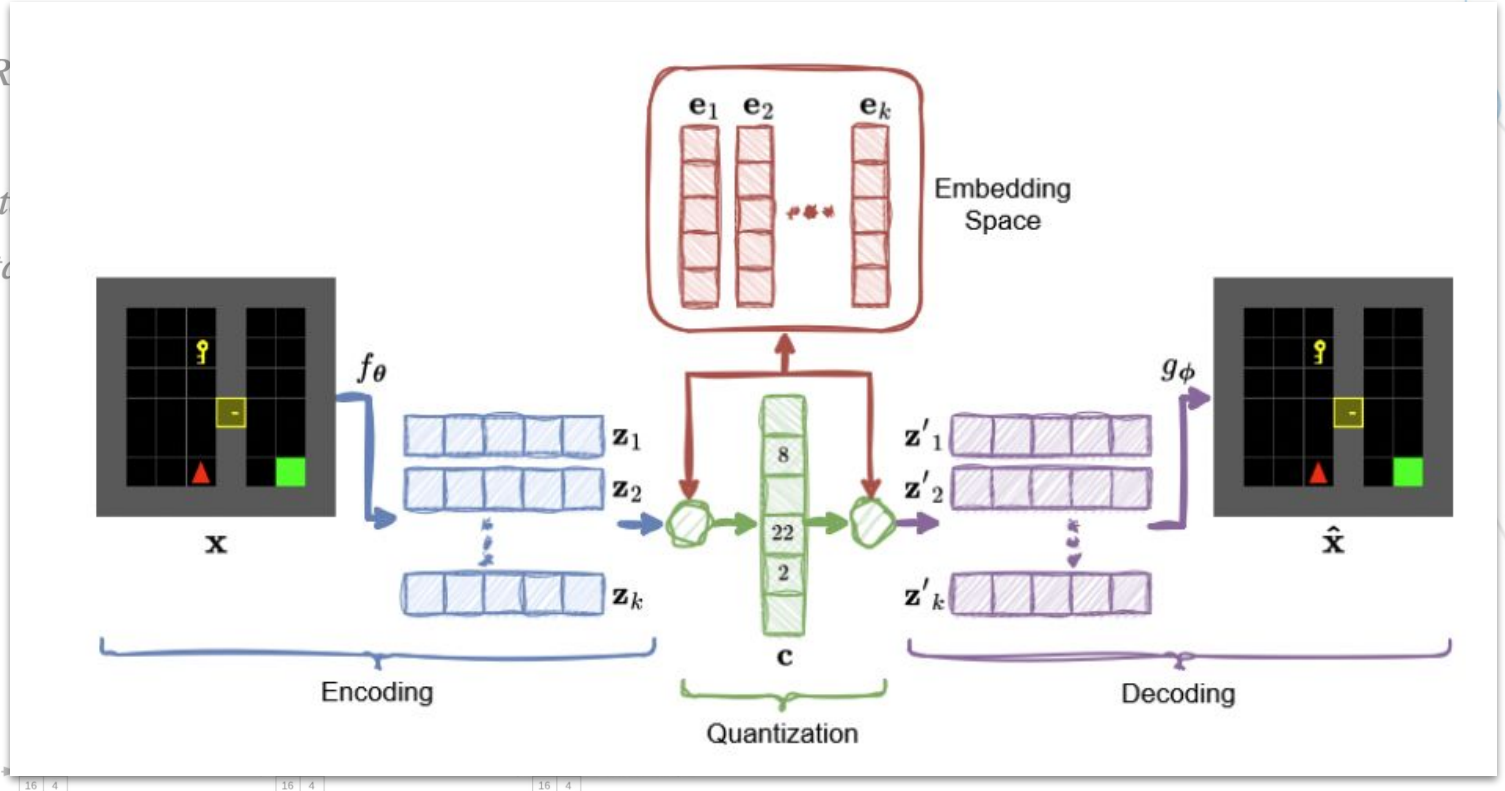
$$
\begin{aligned}
\textit{Representation} \quad & s_t^0 = h(o_{\leq t}) \\
\textit{Prediction} \quad & p_t^k, v_t^k = f(s_t^k) \\
\textit{Afterstate Dynamics} \quad & as_t^k = \phi(s_t^k, a_{t+k}) \\
\textit{Afterstate Prediction} \quad & \sigma_t^k, Q_t^k = \psi(as_t^k) \\
\textit{Dynamics} \quad & s_t^{k+1}, r_t^{k+1} = g(as_t^k, c_{t+k+1})
\end{aligned}
$$

VQ-VAE tricks





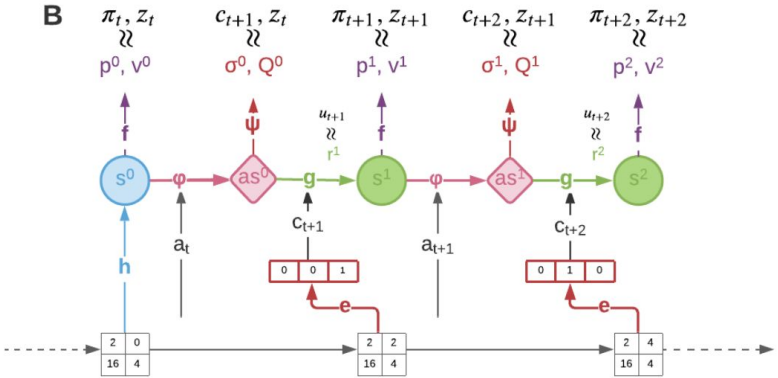
Stochastic MuZero [Antonoglou et al., 2022]

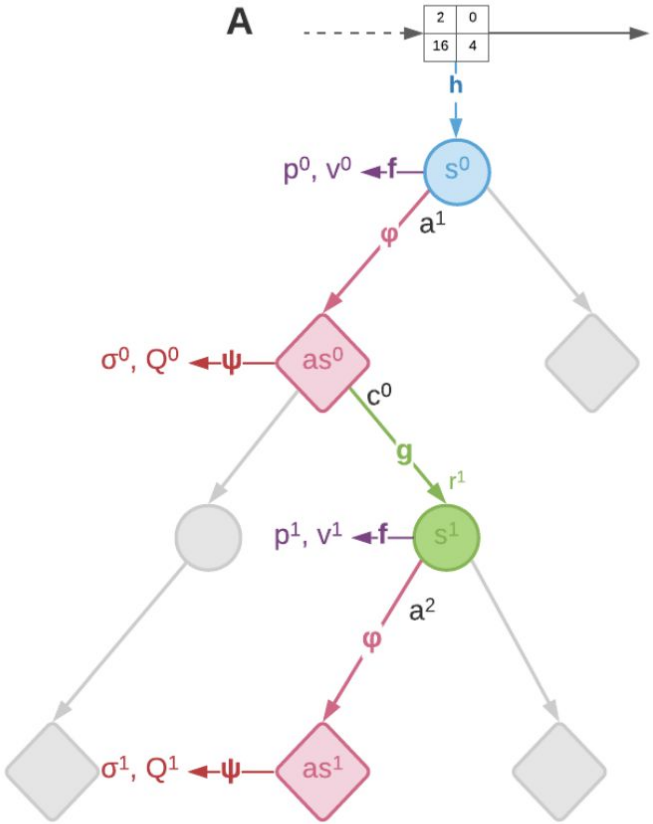


Stochastic MuZero [Antonoglou et al., 2022]

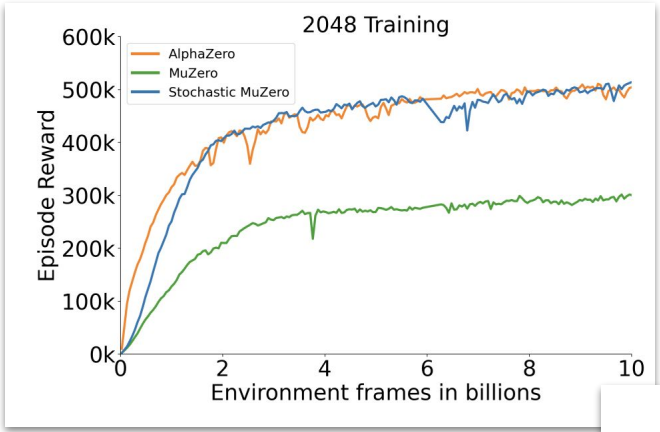
$$
\begin{aligned}
\textit{Representation} \quad & s_t^0 = h(o_{\leq t}) \\
\textit{Prediction} \quad & p_t^k, v_t^k = f(s_t^k) \\
\textit{Afterstate Dynamics} \quad & as_t^k = \phi(s_t^k, a_{t+k}) \\
\textit{Afterstate Prediction} \quad & \sigma_t^k, Q_t^k = \psi(as_t^k) \\
\textit{Dynamics} \quad & s_t^{k+1}, r_t^{k+1} = g(as_t^k, c_{t+k+1})
\end{aligned}
$$

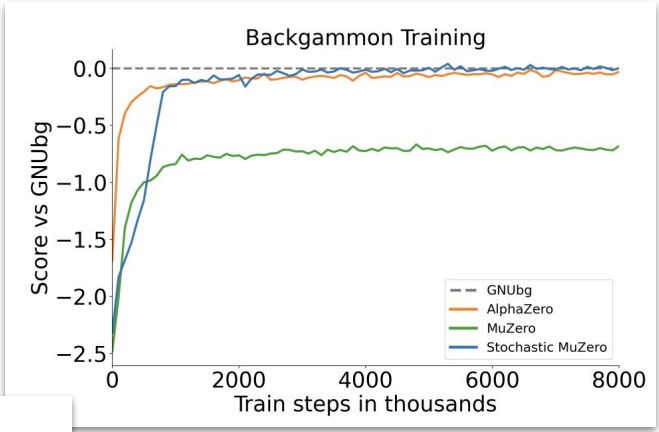
VQ-VAE tricks

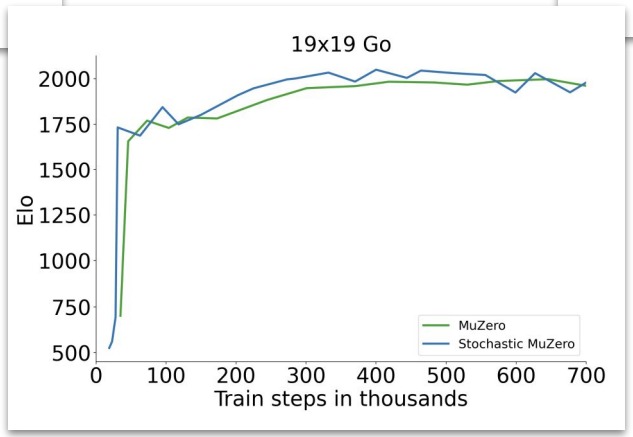




Stochastic MuZero [Antonoglou et al., 2022]







Sampled MuZero [Hubert et al., 2021]

- There have also been MuZero extensions for continuous action spaces
- The underlying idea is to sample actions instead of enumerating all possible actions (and many other details that come with it)

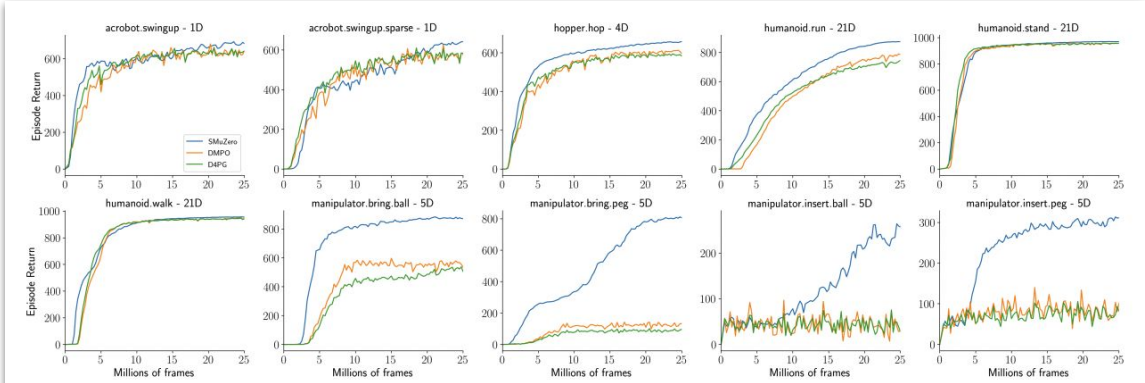
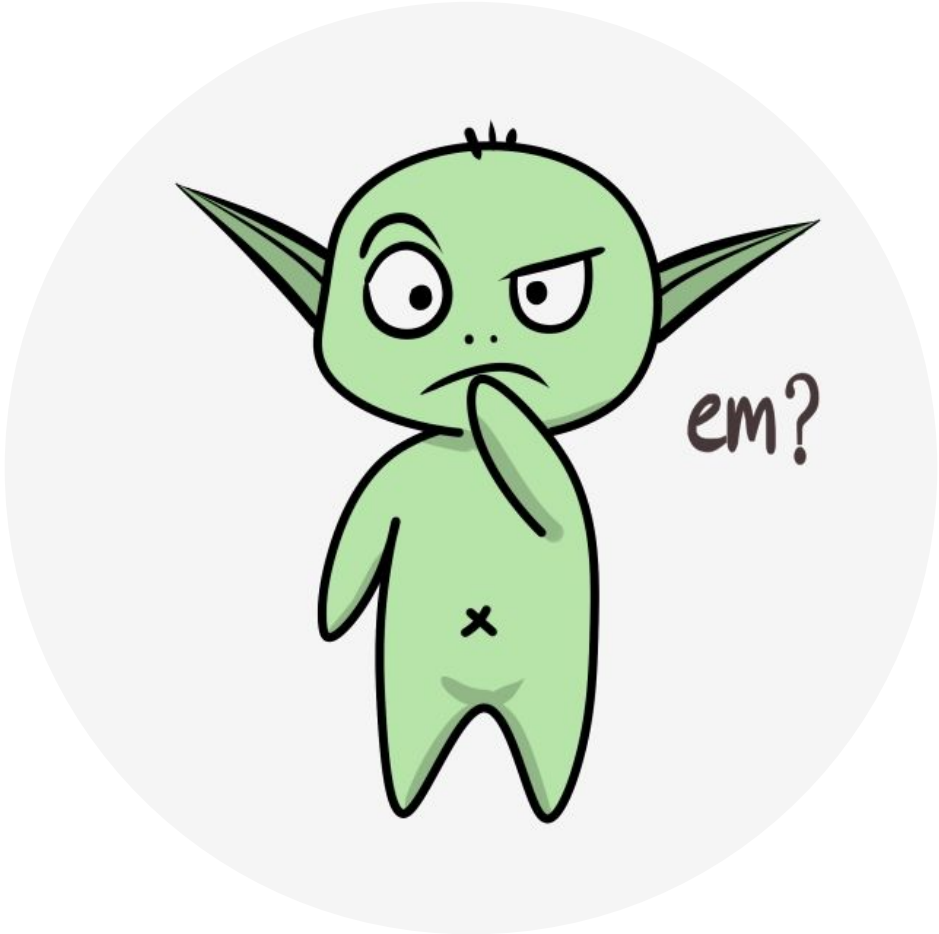


Figure 4. **Results in DM Control Suite Hard and Manipulator tasks.** Performance of *Sampled MuZero* (3 seeds per experiment) throughout training compared to DMPO (Hoffman et al., 2020) and D4PG (Barth-Maron et al., 2018). The x-axis shows millions of environment frames, the y-axis mean episode return. Hard tasks as proposed by (Hoffman et al., 2020). Plot titles include the task name and the dimensionality of the action space.



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Thank you!