Planning in deep reinforcement learning

Alexander Lyzhov

MDP

Planning

Planning in deep RL

MuZero architecture

MuZero results

MDP

Trajectory: $s_0, a_1, r_1, s_1, a_2, r_2, s_2, a_3, r_3, s_3, \dots$

Dynamics: $\mathbb{P}(s', r \mid s, a)$
~~Model~~ policy: $\pi(a \mid s)$

Return: $\underline{g_t} = r_{t+1} + \overset{<1}{\gamma}\, \underline{r_{t+2} + \gamma^2 r_{t+3} + \gamma_3 r_{t+4} + \cdots}$
Value function: $v_\pi(s) = \mathbb{E}_\pi(g_t \mid s_t = s)$
Q-value function: $q_\pi(s, a) = \mathbb{E}_\pi(g_t \mid s_t = s, a_t = a)$

$$g_t = r_{t+1} + \gamma g_{t+1}$$
$$v_\pi(s) = \mathbb{E}_{a_{t+1}, r_{t+1}, s_{t+1}}(r_{t+1} + \gamma v_\pi(s_{t+1}) \mid s_t = s)$$
$$q_\pi(s, a) = \mathbb{E}_{r_{t+1}, s_{t+1}, a_{t+2}}(r_{t+1} + \gamma q_\pi(s_{t+1}, a_{t+2}) \mid s_t = s, a_{t+1} = a)$$

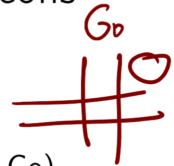
Planning

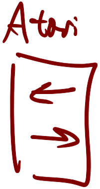
Environment model - anything that can be used to predict how the env responds to actions.
Planning - anything that improves policy with a model.

- Background planning - to improve the policy when training
- Online planning - to make better decision with trained policy

Model-based vs model-free - continuum.
Analogies: self-supervision, augmentation, actor-critic methods, 2-player game [1].

Planning: pros and cons

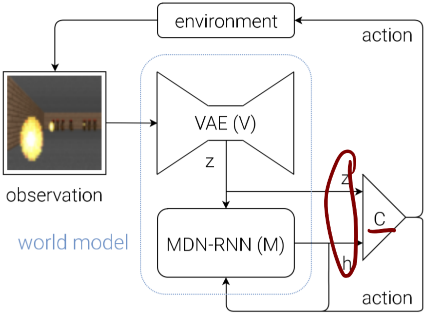




+ works in domains with complex dynamics (e.g. Go)
+ more sample-efficient
+ better transfer, including zero-shot adaptation to new tasks [2] plan2explore
+ better offline RL [3] $p: s, a \rightarrow r$
− takes time/memory to build environment model and use it
− more complex architecture, more hyperparameters

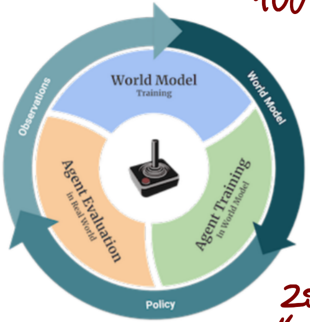
Model-based deep RL in discrete environments

World models[4]



SimPLe [5]

400K



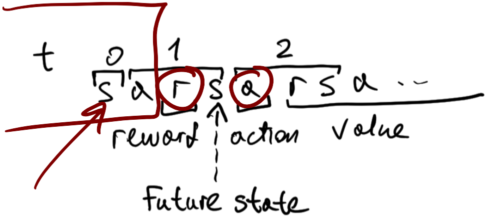
2017

Also: Value Iteration Networks [6], Value Prediction Networks [7], Predictron [8].
But SOTA was model-free before MuZero (R2D2 on Atari).

MRP

MuZero[9]

Discrete state & action spaces, deterministic dynamics: Atari, Go, Chess, Shogi.
Idea: only predict those aspects of the future that help to pick the best action.



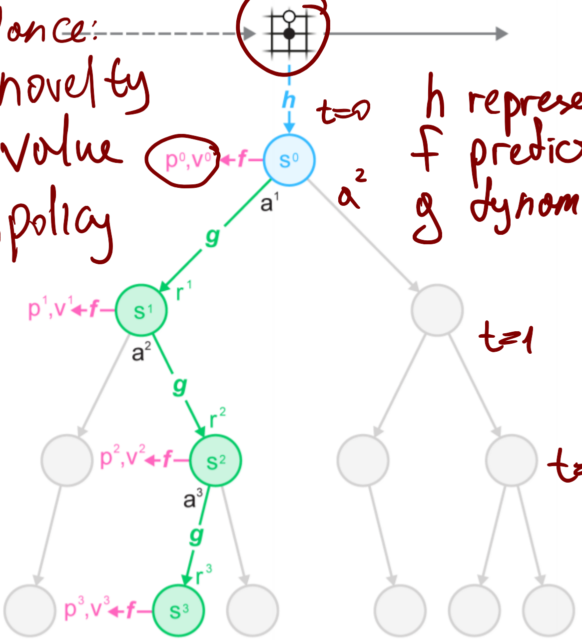
Want to predict:

policy: $\pi(a_{t+k+1} \mid o_{1..t}, a_{1..t+k})$ $\forall k$

value: $\mathbb{E}(r_{t+k+1} + \gamma r_{t+k+2} + \dots \mid o_{1..t}, a_{1..t+k})$

reward: $p(r_{t+k} \mid o_{1..t}, a_{1..t+k})$

balance:
1) novelty
2) value
3) policy



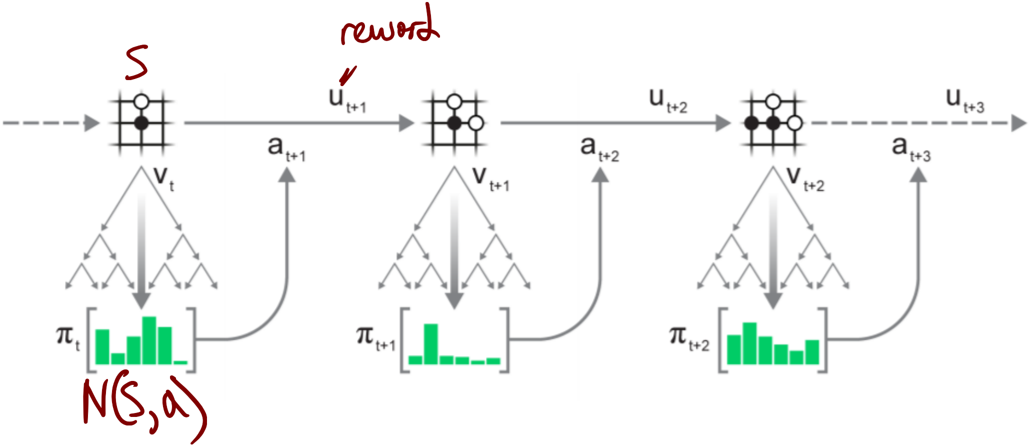
h representation
f prediction
g dynamics

MuZero planning: MCTS

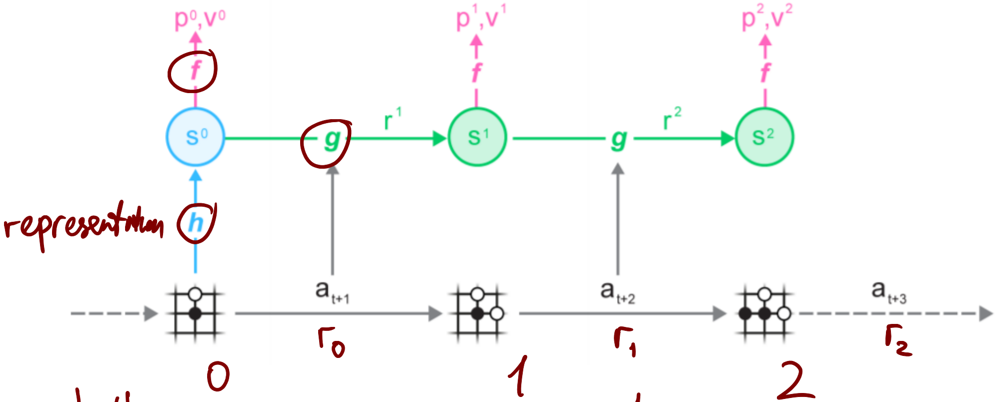
1) Repeatedly select actions by balancing exploration with exploitation on each timestep
2) Expand tree with leaf state, remember final reward and transition
3) Update values $Q(s, a)$ on trajectory using final value and intermediate rewards

Result: visit counts $N(s_0, a)$

MuZero acting



MuZero training



tree depth reward values policy

$$l_t(\theta) = \sum_{k=0}^{K} l^r\left(r_{t+k}, r_t^k\right) + l^v\left(z_{t+k}, v_t^k\right) + l^p\left(\pi_{t+k}, \mathbf{p}_t^k\right) + c\|\theta\|^2$$

$$z_t = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n v_{t+n}$$

MCTS result policy inside MCTS

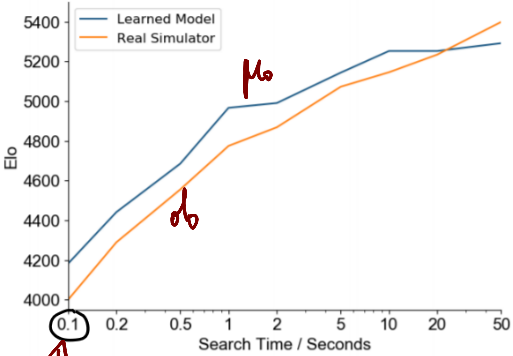
Hyperparameters

16 blocks

- Deep resnets for dynamics g and representation h, shallow convolutional net with FC for prediction module f
- Planning horizon K = 5 — layers in tree for backprop
- Bootstrapping values: 10 steps for Atari, to end of game for board games
- 1M mini-batches of size 1-2K. Prioritized replay for Atari, uniform replay for board games

MuZero results

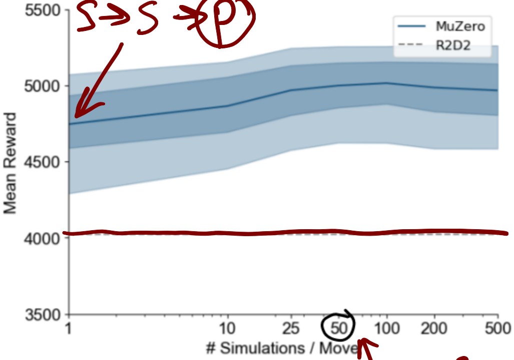
Go



800 rollouts

real latent Atari

S → S → P



MCTS rollouts

$$\frac{S_{model}-S_{random}}{S_{human}-S_{random}}$$

Human normalized scores

Agent	Median	Mean	Env. Frames
Ape-X	434.1%	1695.6%	22.8B
R2D2	1920.6%	4024.9%	37.5B
MuZero	**2041.1%**	**4999.2%**	20.0B
IMPALA	191.8%	957.6%	200M
Rainbow	231.1%	–	200M
UNREAL	250%[a]	880%[a]	250M
LASER	431%	–	200M
MuZero Reanalyze	**731.1%**	**2168.9%**	200M

Atari
20TPU: 12h

400K

References I

Aravind Rajeswaran, Igor Mordatch, and Vikash Kumar. "A game theoretic framework for model based reinforcement learning". In: *arXiv preprint arXiv:2004.07804* (2020).

Ramanan Sekar et al. "Planning to Explore via Self-Supervised World Models". In: *arXiv preprint arXiv:2005.05960* (2020).

Tianhe Yu et al. *MOPO: Model-based Offline Policy Optimization*. 2020. arXiv: `2005.13239 [cs.LG]`.

David Ha and Jürgen Schmidhuber. "World models". In: *arXiv preprint arXiv:1803.10122* (2018).

Lukasz Kaiser et al. "Model-based reinforcement learning for atari". In: *arXiv preprint arXiv:1903.00374* (2019).

References II

Aviv Tamar et al. "Value iteration networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 2154–2162.

Junhyuk Oh, Satinder Singh, and Honglak Lee. "Value prediction network". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6118–6128.

David Silver et al. "The predictron: End-to-end learning and planning". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 3191–3199.

David Silver et al. "Mastering chess and shogi by self-play with a general reinforcement learning algorithm". In: *arXiv preprint arXiv:1712.01815* (2017).

Julian Schrittwieser et al. "Mastering atari, go, chess and shogi by planning with a learned model". In: *arXiv preprint arXiv:1911.08265* (2019).

References III

Danijar Hafner et al. “Dream to control: Learning behaviors by latent imagination”. In: *arXiv preprint arXiv:1912.01603* (2019).