Thinking Fast and Slow with Deep Learning and Tree Search

by
Thomas Anthony, Zheng Tian, David Barber
(University College London)

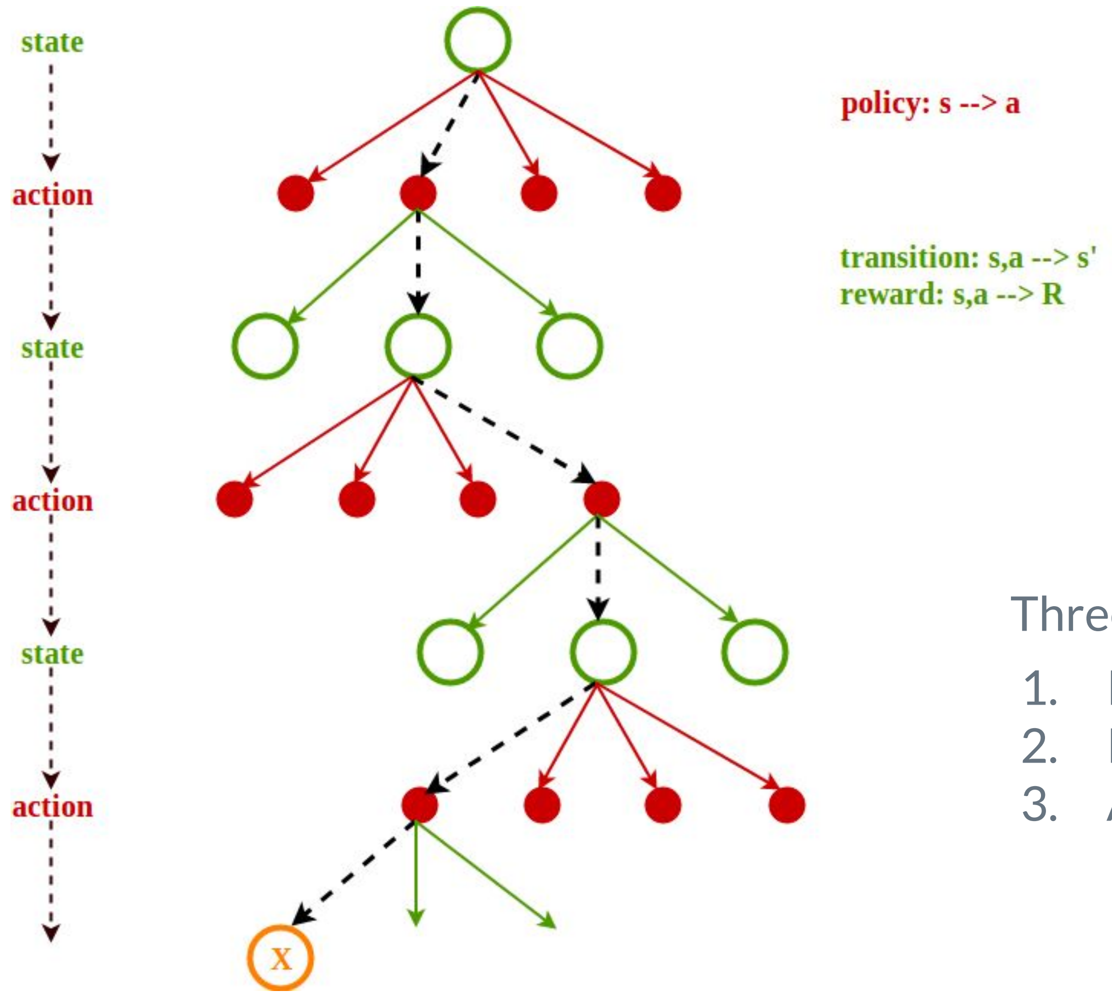
NIPS Debriefing

Thomas Moerland
(TU Delft)

Content

1. Background: Search & Reinforcement Learning
2. Algorithm: Expert Iteration (ExIt) → Iterating Search & RL
3. Results: Hex (game)
4. Discussion: Interpretation

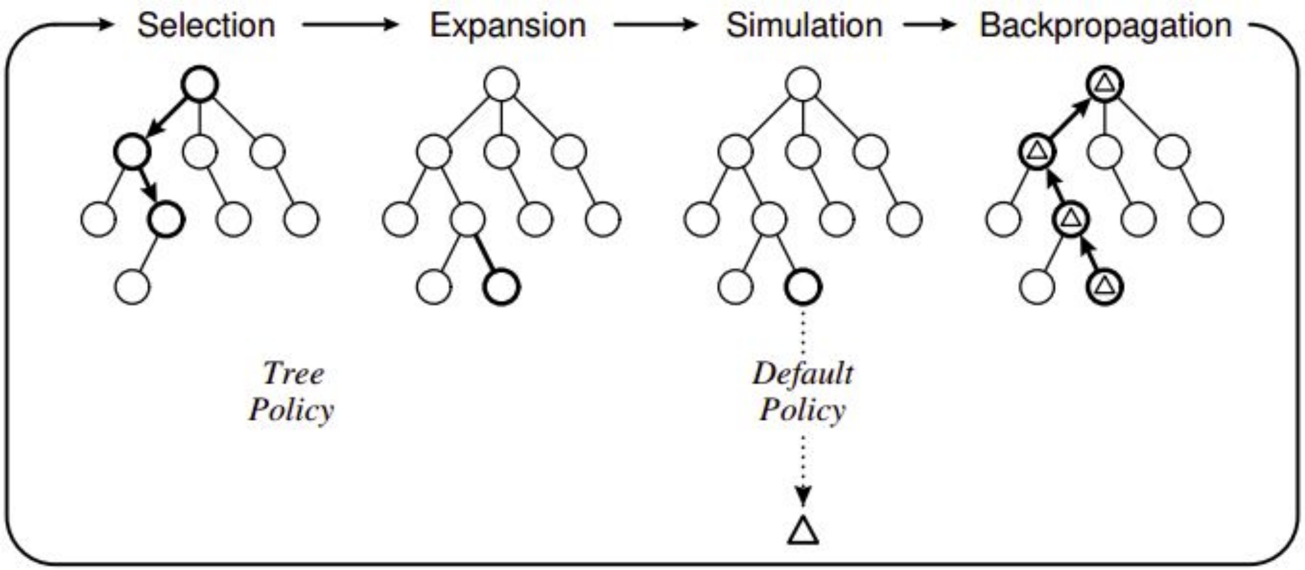
Sequential Decision Making



Three goals:

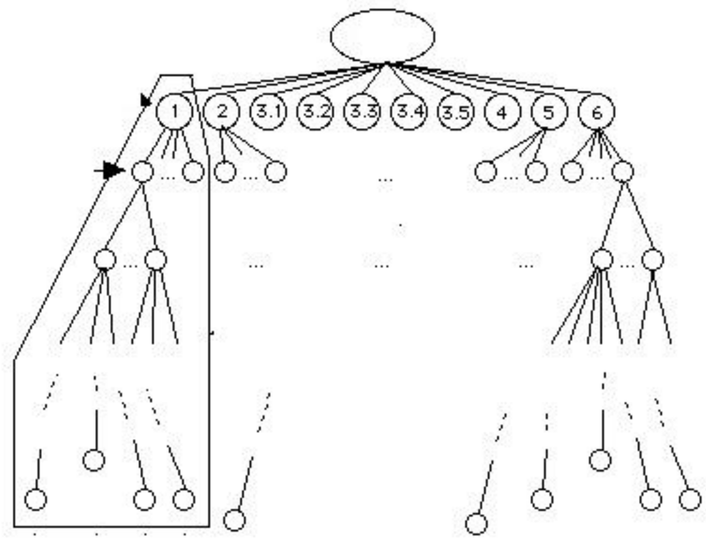
1. Reduce breadth
2. Reduce depth
3. Avoid repeating work

Monte Carlo Tree Search



+ Keep information local
+ Uncertainty for exploration

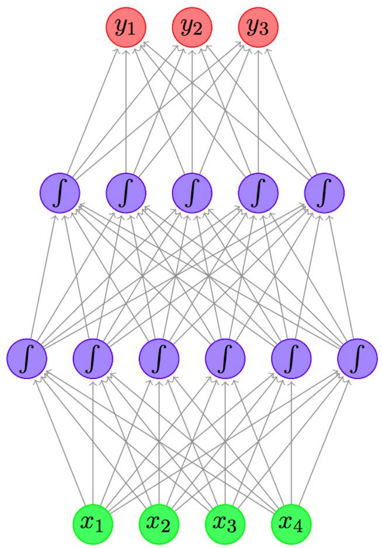
Monte Carlo Tree Search

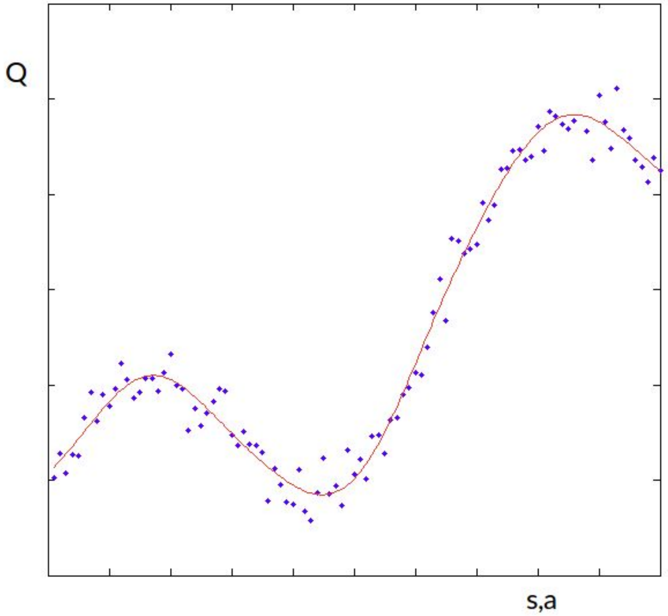


+ Keep information local
+ Uncertainty for exploration

- No generalization
- Does not scale (memory)

Reinforcement Learning

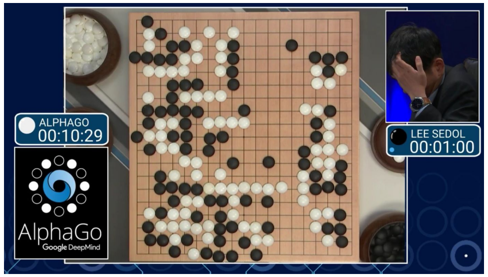


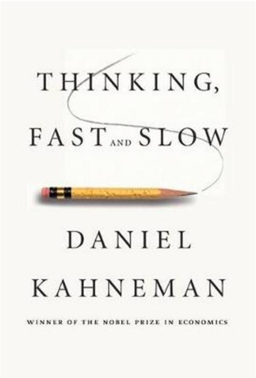


+ Generalization
+ Bootstrapping

- Unstable / local minima / high variance
- Poor exploration
- (single trace)

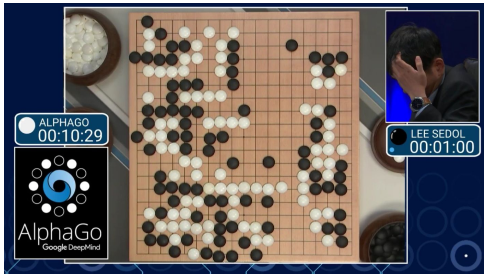
Algorithm: Expert Iteration (ExIt)

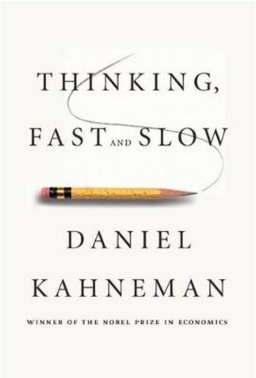




Tree Search	**Supervised Learning (RL)**
'Thinking slow' *Expert*	'Thinking fast' *Apprentice*

Algorithm: Expert Iteration (ExIt)

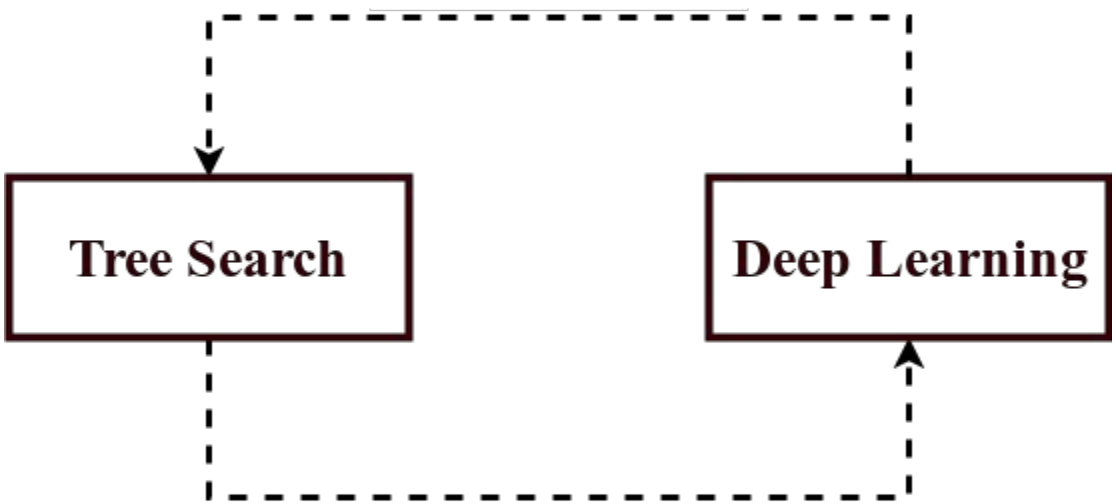




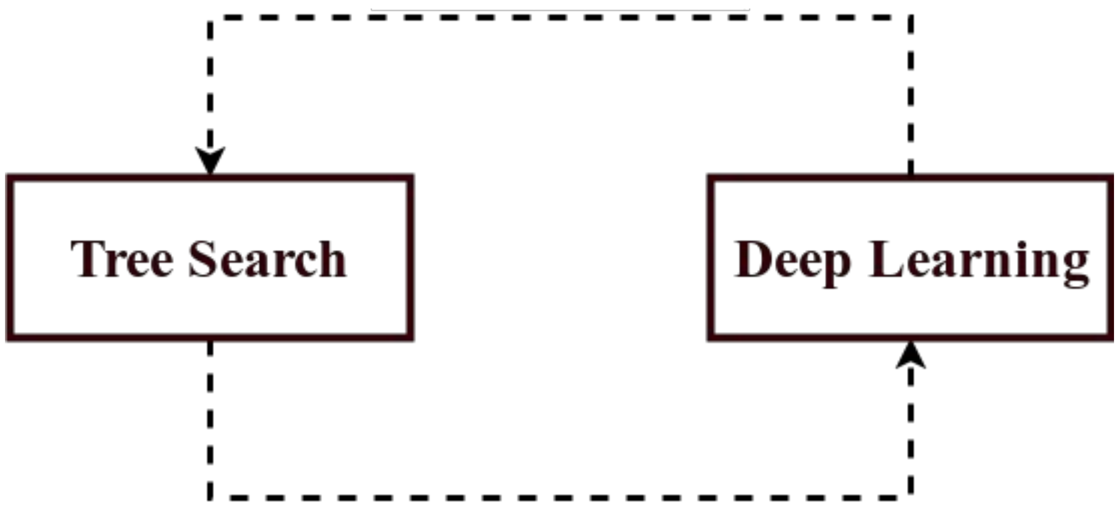
Tree Search	Supervised Learning (RL)
‘Thinking slow’ *Expert*	‘Thinking fast’ *Apprentice*

=

Iterating (local) search & generalization

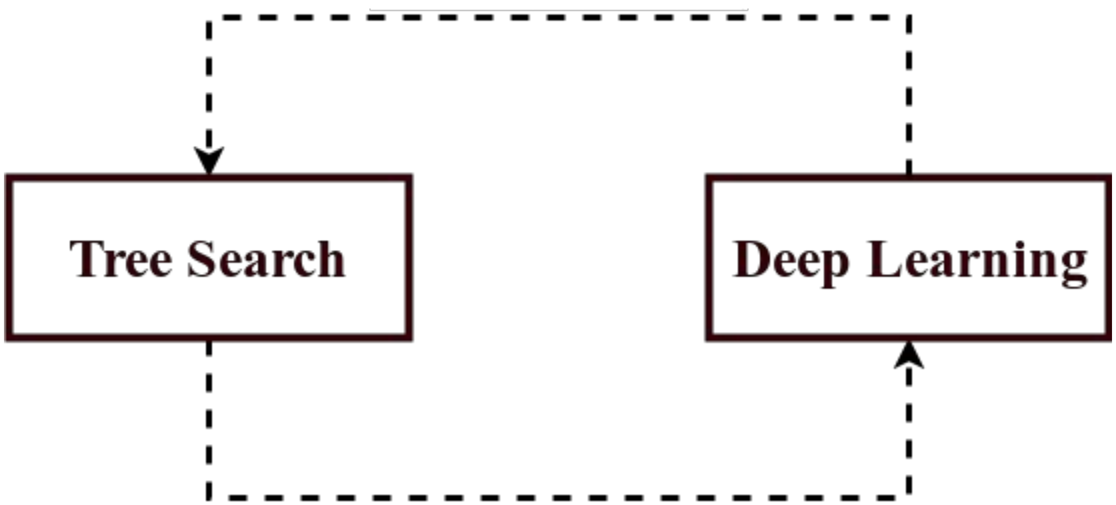


Iterating (local) search & generalization



What to store?

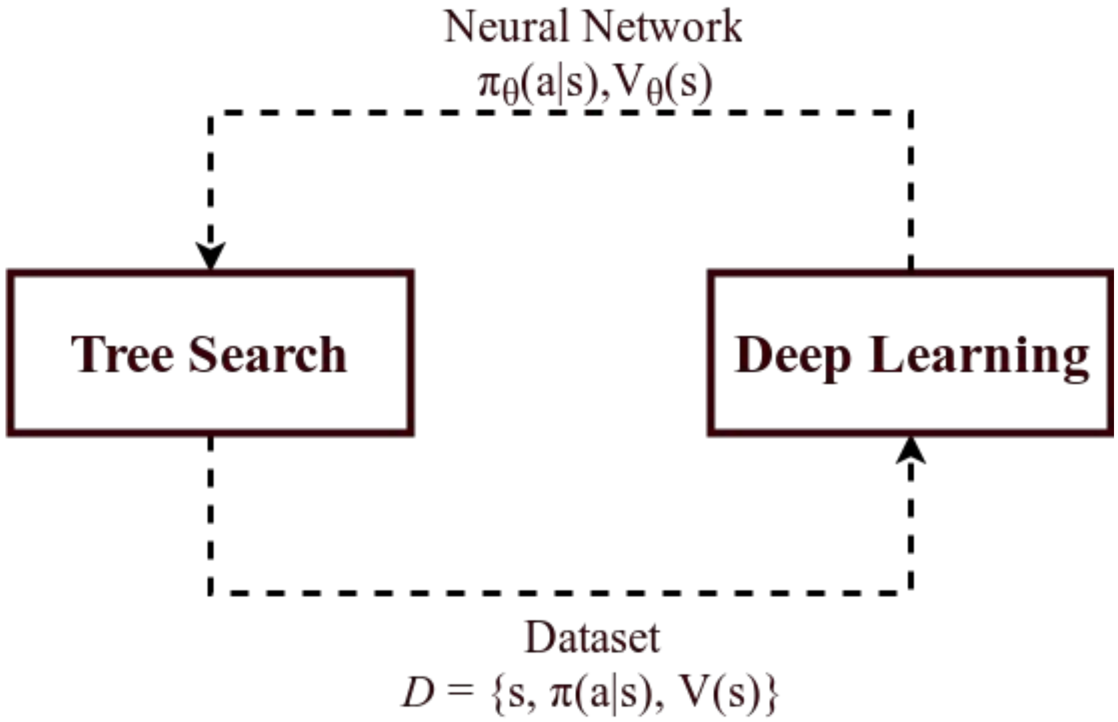
Iterating (local) search & generalization



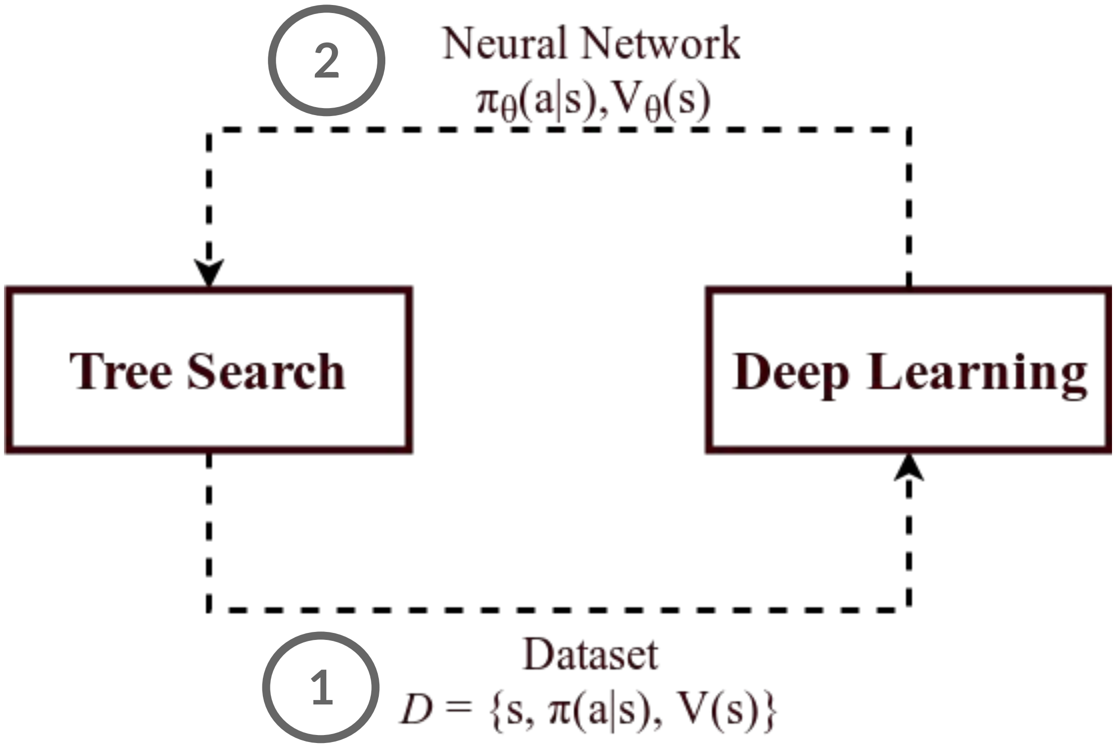
<u>What to store?</u>

I. Reduce breadth	*Policy*	$s \longrightarrow \pi(a)$
II. Reduce depth	*Value*	$s \longrightarrow V$

Iterating (local) search & generalization

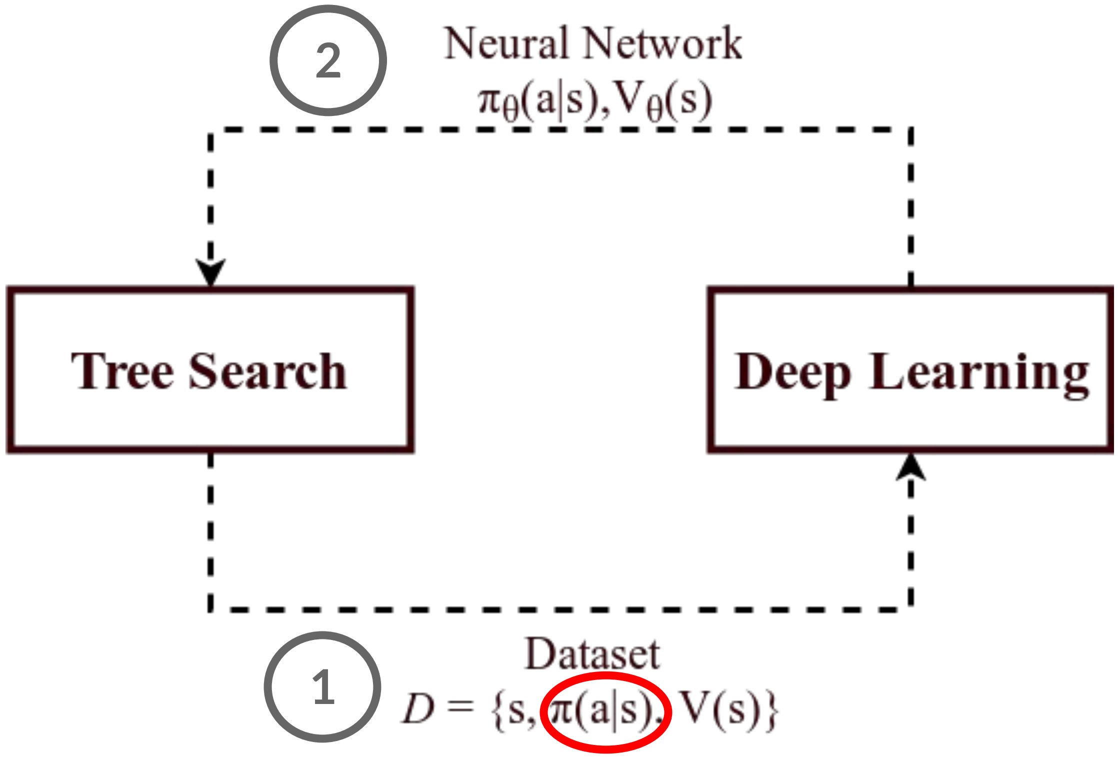


Iterating (local) search & generalization



1. How to process the tree search information?
2. How to use the network in the tree search?

Iterating (local) search & generalization



1. **How to process the tree search information?**
2. How to use the network in the tree search?

How to process the tree search result?

Policy: MCTS picks action with highest $n(s,a)$ at the root.

How to process the tree search result?

Policy: MCTS picks action with highest n(s,a) at the root.

1. Chosen-action target (CAT):

$$a^* = \text{argmax}_a(n(s,a)) \qquad \mathcal{L}_{\text{CAT}} = -\log[\pi(a^*|s)]$$

How to process the tree search result?

Policy: MCTS picks action with highest n(s,a) at the root.

1. Chosen-action target (CAT):

$$a^* = \mathrm{argmax}_a(n(s,a)) \qquad \mathcal{L}_{\mathrm{CAT}} = -\log[\pi(a^*|s)]$$

2. Tree-policy Targets (TPT):

$$\pi^*(a|s) = \frac{n(s,a)}{n(s)} \qquad \mathcal{L}_{\mathrm{TPT}} = -\sum_a \frac{n(s,a)}{n(s)} \log[\pi(a|s)]$$

How to process the tree search result?

Policy: MCTS picks action with highest n(s,a) at the root.

1. Chosen-action target (CAT):

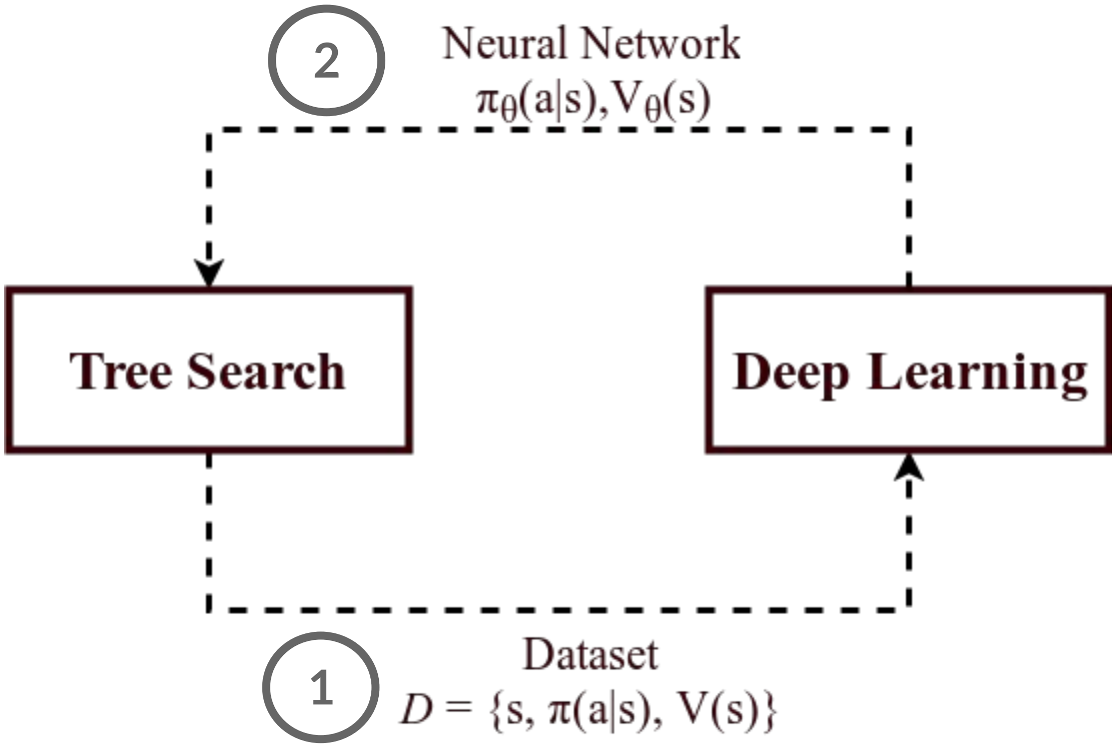
$$a^* = \mathrm{argmax}_a(n(s,a)) \qquad \mathcal{L}_{\mathrm{CAT}} = -\log[\pi(a^*|s)]$$

2. Tree-policy Targets (TPT):

$$\pi^*(a|s) = \frac{(n(s,a))^{T}}{\sum(n(s,a))^{T}} \qquad \mathcal{L}_{\mathrm{TPT}} = -\sum_a \frac{n(s,a)}{n(s)} \log[\pi(a|s)]$$

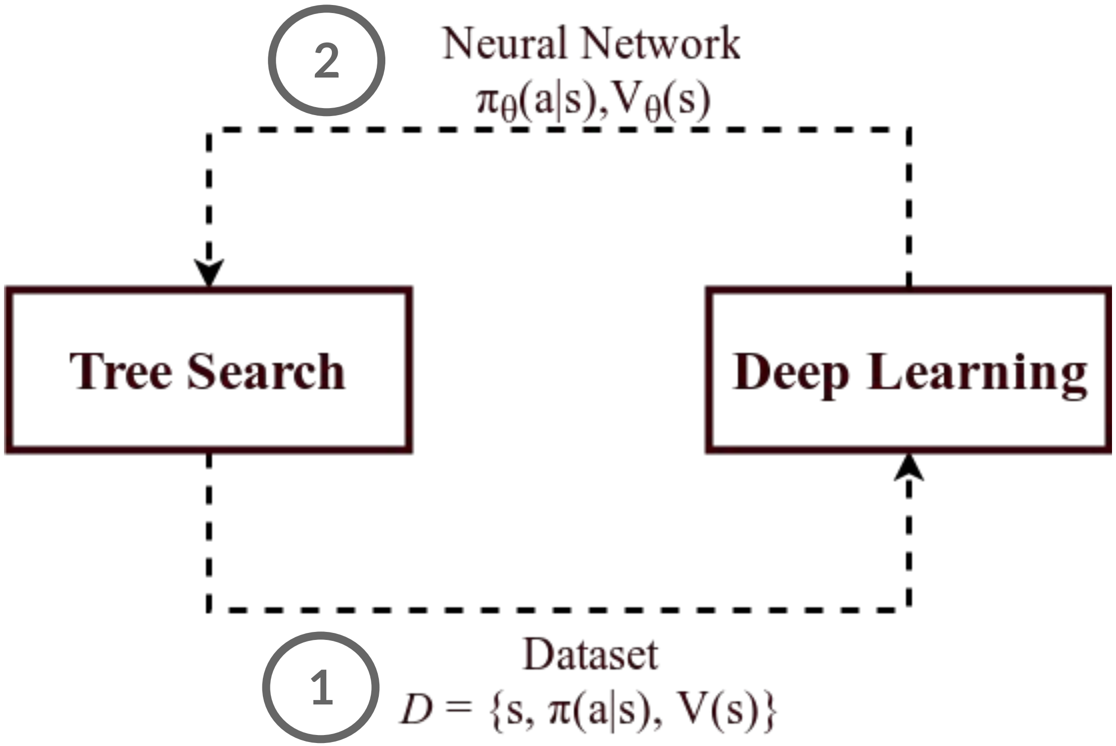
Silver, David, et al. "Mastering the game of go without human knowledge." *Nature* 550.7676 (2017): 354.

Iterating (local) search & generalization



1. How to process the tree search information?
2. How to use the network in the tree search?

Iterating (local) search & generalization



1. How to process the tree search information?
2. **How to use the network in the tree search?**

How to warm start the tree search?

Policy (reduce breadth):

$$\mathrm{UCT}(s,a) = \frac{r(s,a)}{n(s,a)} + c_b\sqrt{\frac{\log n(s)}{n(s,a)}}$$

How to warm start the tree search?

Policy: This paper (ExIt)

$$\mathrm{UCT}(s,a) = \frac{r(s,a)}{n(s,a)} + c_b\sqrt{\frac{\log n(s)}{n(s,a)}} + w_a \frac{\hat{\pi}(a|s)}{n(s,a)+1}$$

How to warm start the tree search?

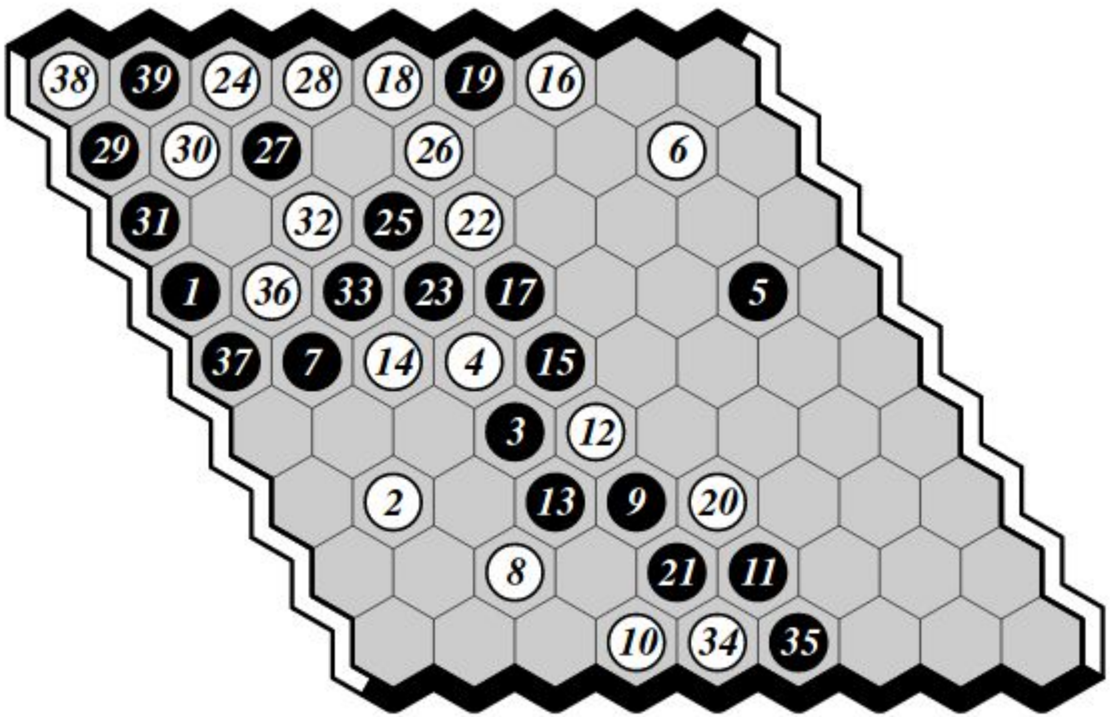
Policy: AlphaGo Zero

$$\mathrm{UCT}(s,a) = \frac{r(s,a)}{n(s,a)} + w_a \frac{\hat{\pi}(a|s)}{n(s,a)+1}$$

Silver, David, et al. "Mastering the game of go without human knowledge." *Nature* 550.7676 (2017): 354.

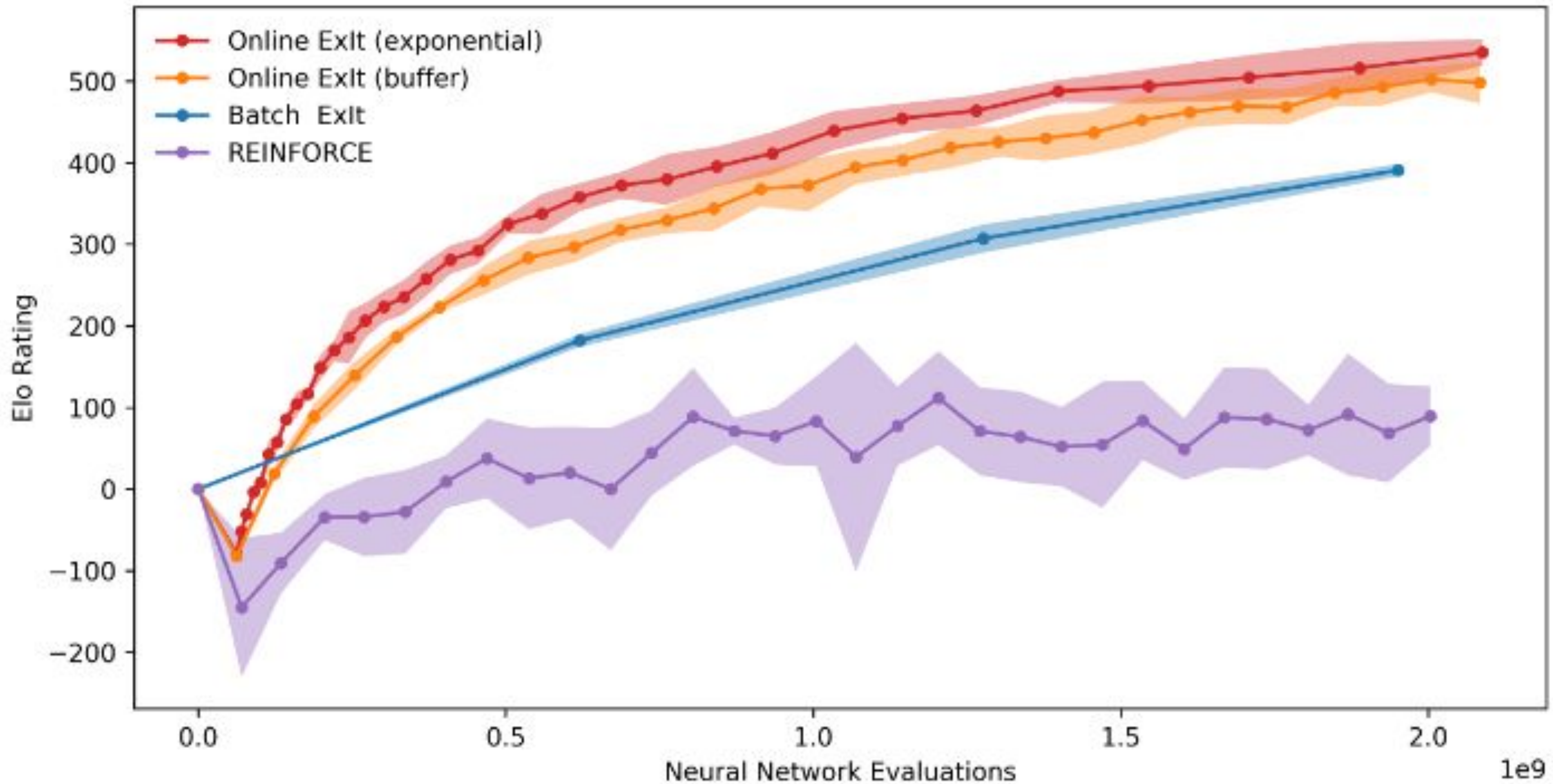
Results

Game = Hex



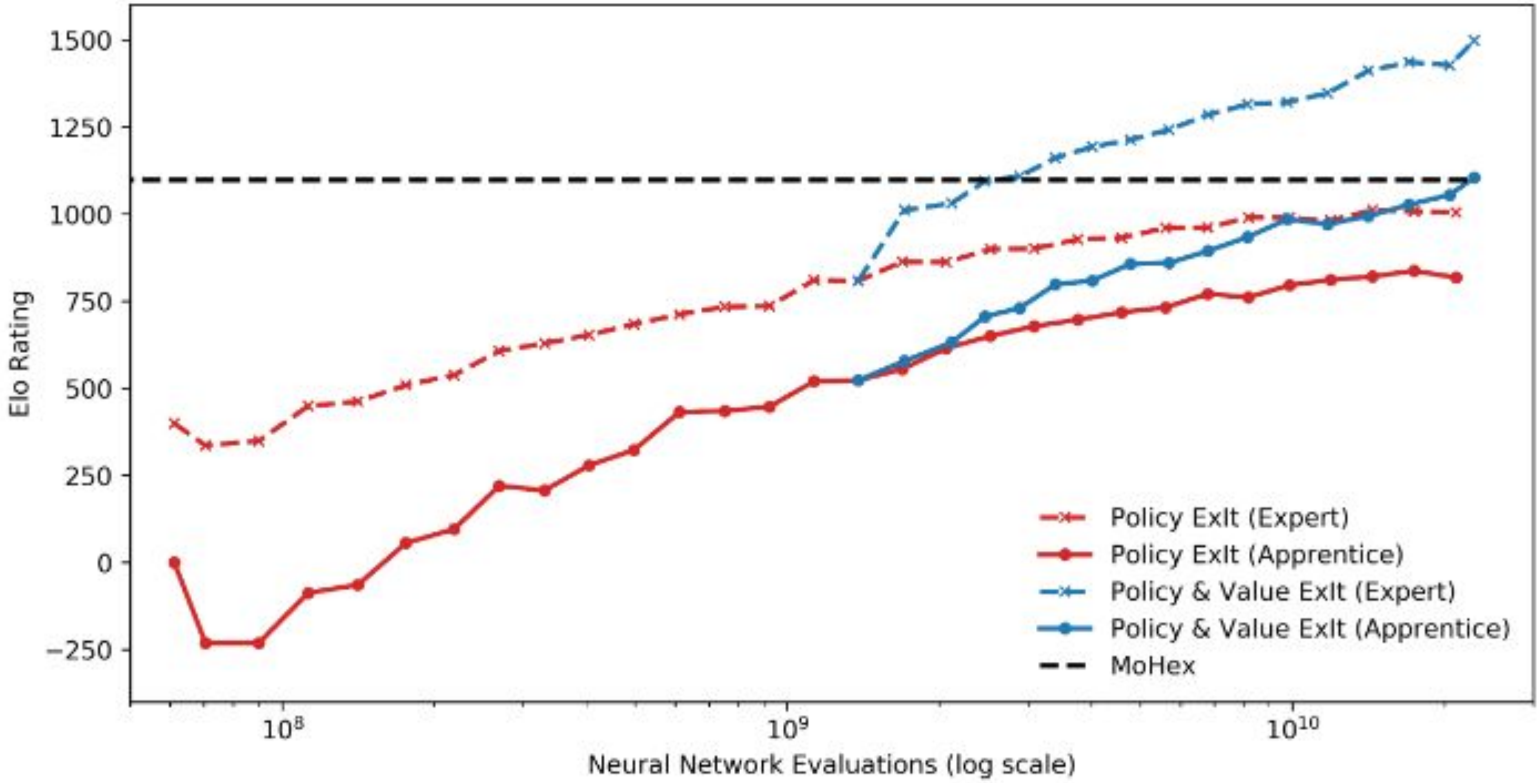
Results

Comparison to vanilla RL (policy network only)



Results

Added benefit of value network



Results

Battle versus MoHex

(search-based, game-specific pruning and end-game solving)

ExIt Setting	Time/move	ExIt win rate	MoHex Setting	Solver	Time/move
10^4 iterations	~ 0.3s	75.3%	10^4 iterations	No	$\sim 0.2s$
10^4 iterations	~ 0.3s	59.3%	10^5 iterations	No	$\sim 2s$
10^4 iterations	~ 0.3s	55.6%	$4s$/move	Yes	$4s$

Results

Battle versus MoHex

(search-based, game-specific pruning and end-game solving)

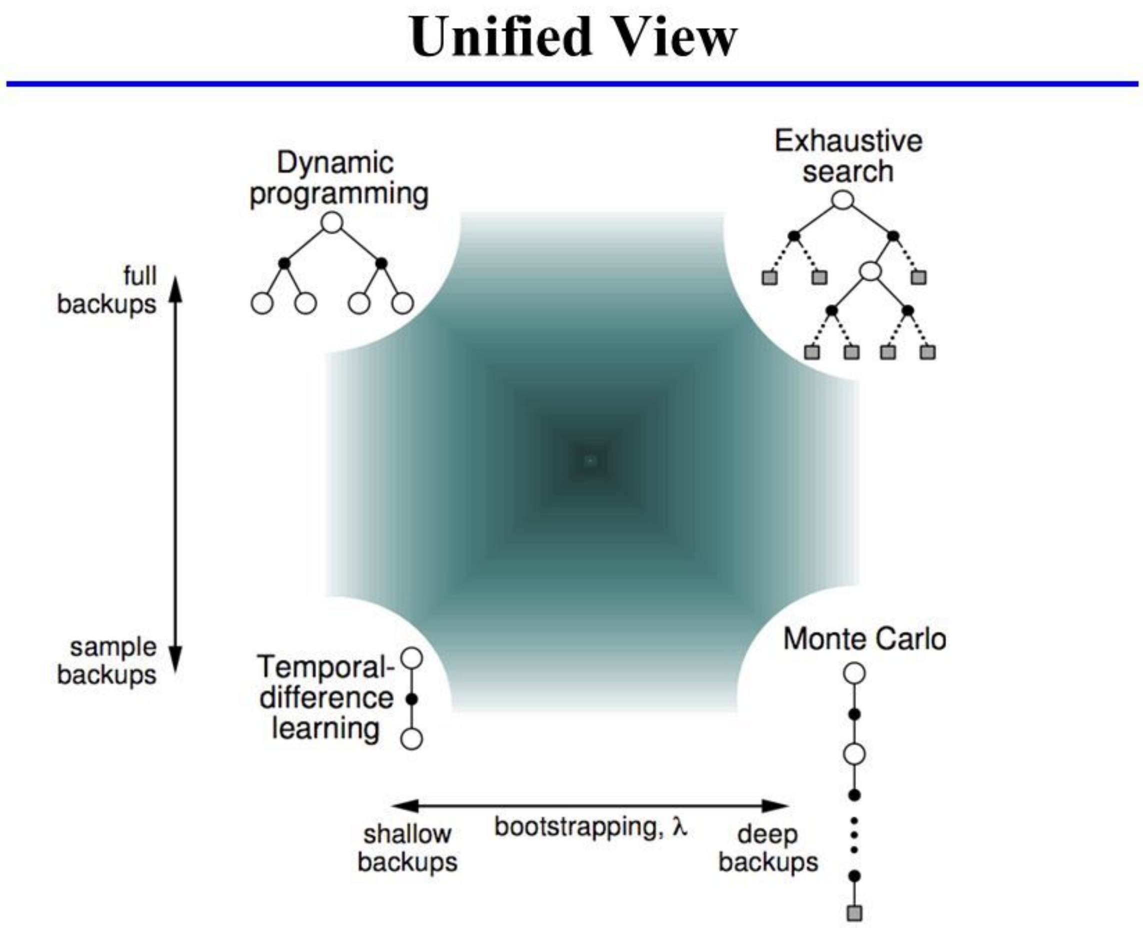
ExIt Setting	Time/move	ExIt win rate	MoHex Setting	Solver	Time/move
10^4 iterations	~ 0.3s	75.3%	10^4 iterations	No	$\sim 0.2s$
10^4 iterations	~ 0.3s	59.3%	10^5 iterations	No	$\sim 2s$
10^4 iterations	~ 0.3s	55.6%	$4s$/move	Yes	$4s$

Maybe not as impressive as AlphaGo Zero

But: Training time ~100.000 times lower than AG0

(AG0 = ~1.6e12 traces)

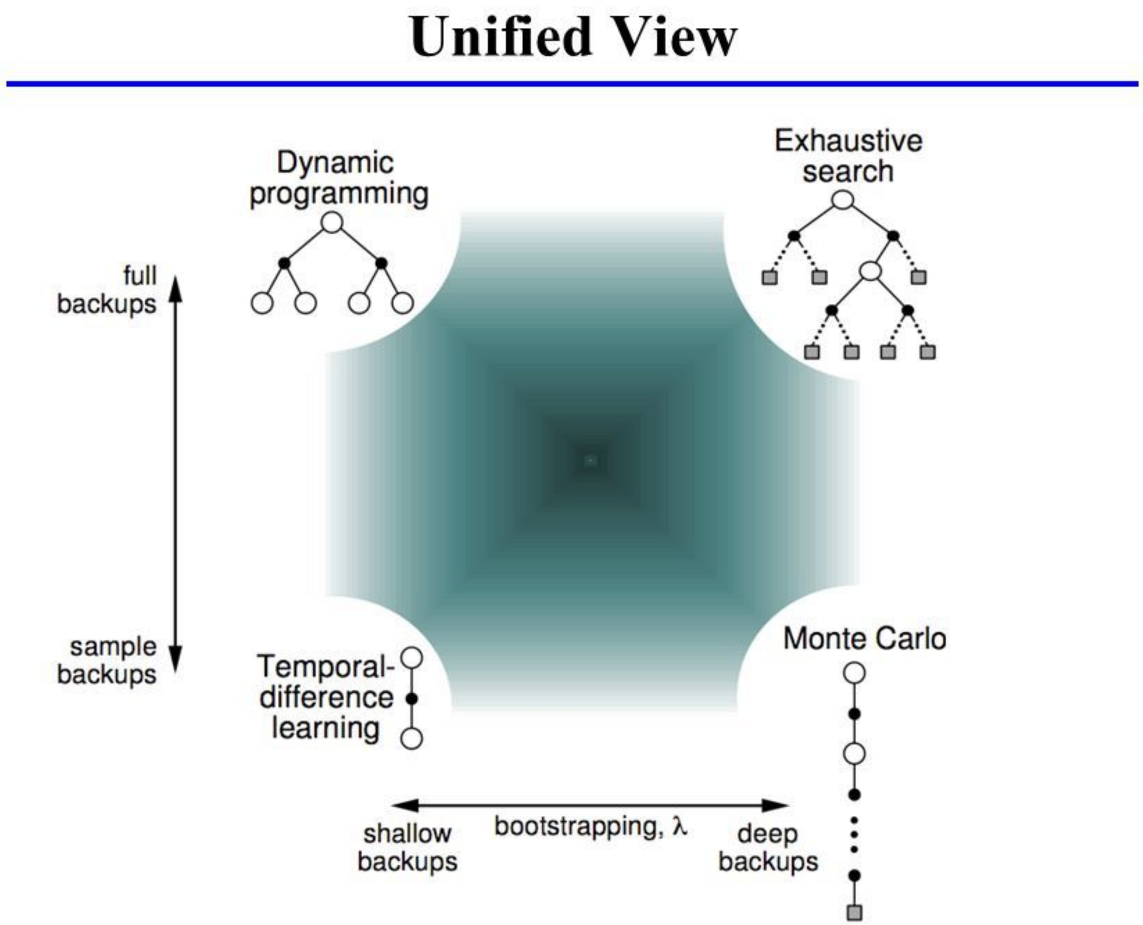
Discussion: Search and RL



Sutton & Barto. Reinforcement Learning: An Introduction. 1998.

Discussion: Search and RL

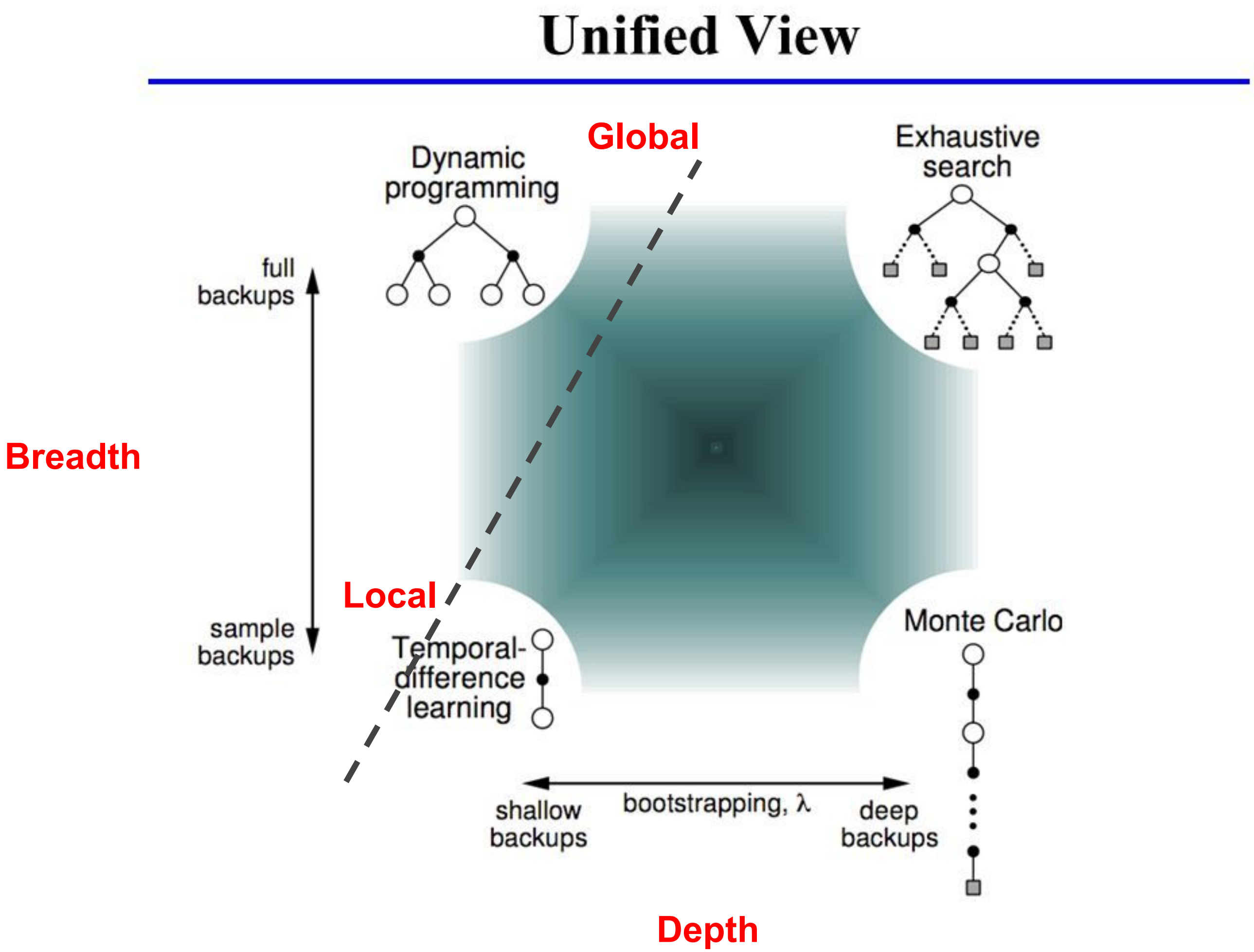
Breadth



Depth

Sutton & Barto. Reinforcement Learning: An Introduction. 1998.

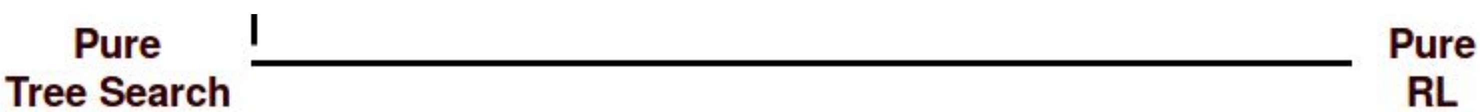
Discussion: Search and RL



Sutton & Barto. Reinforcement Learning: An Introduction. 1998.

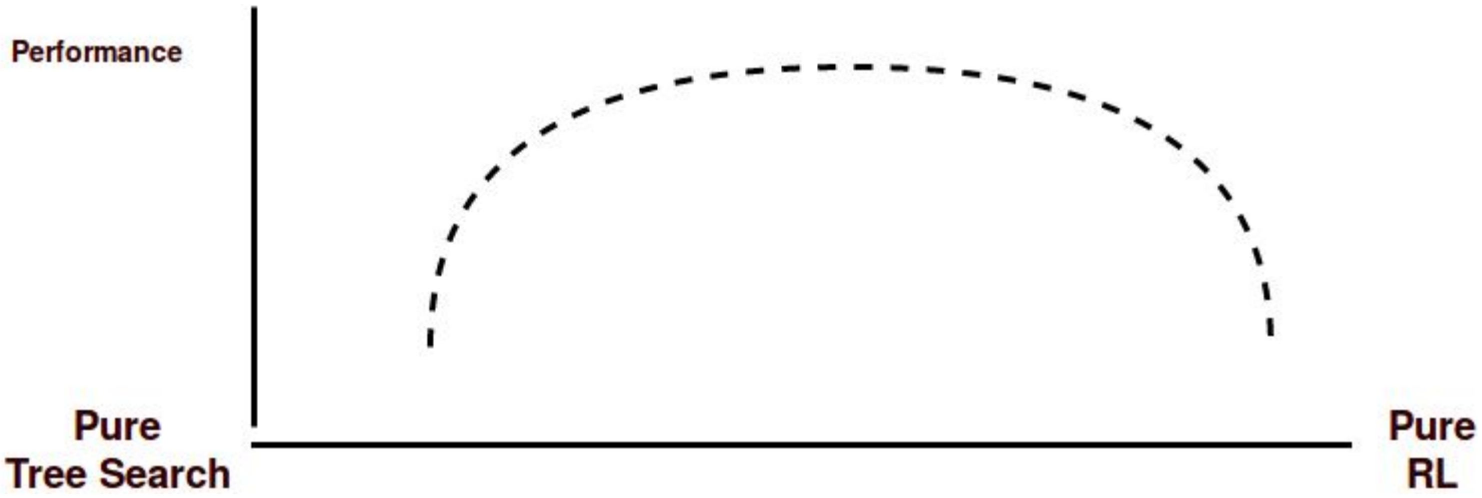
Discussion

Balancing local differentiation and global generalization



Discussion

Balancing local differentiation and global generalization



Discussion

- I empirically observed the problem of too early generalization in RL.

Discussion

- I empirically observed the problem of too early generalization in RL.

Open Questions:

- *Tree search*
- *What to store?*
- *How to steer search from a NN?*
- *Balance search/function approximation*

Thanks!