MuZero

Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model

NSE ML+Security Reading Group

Kim Hammar

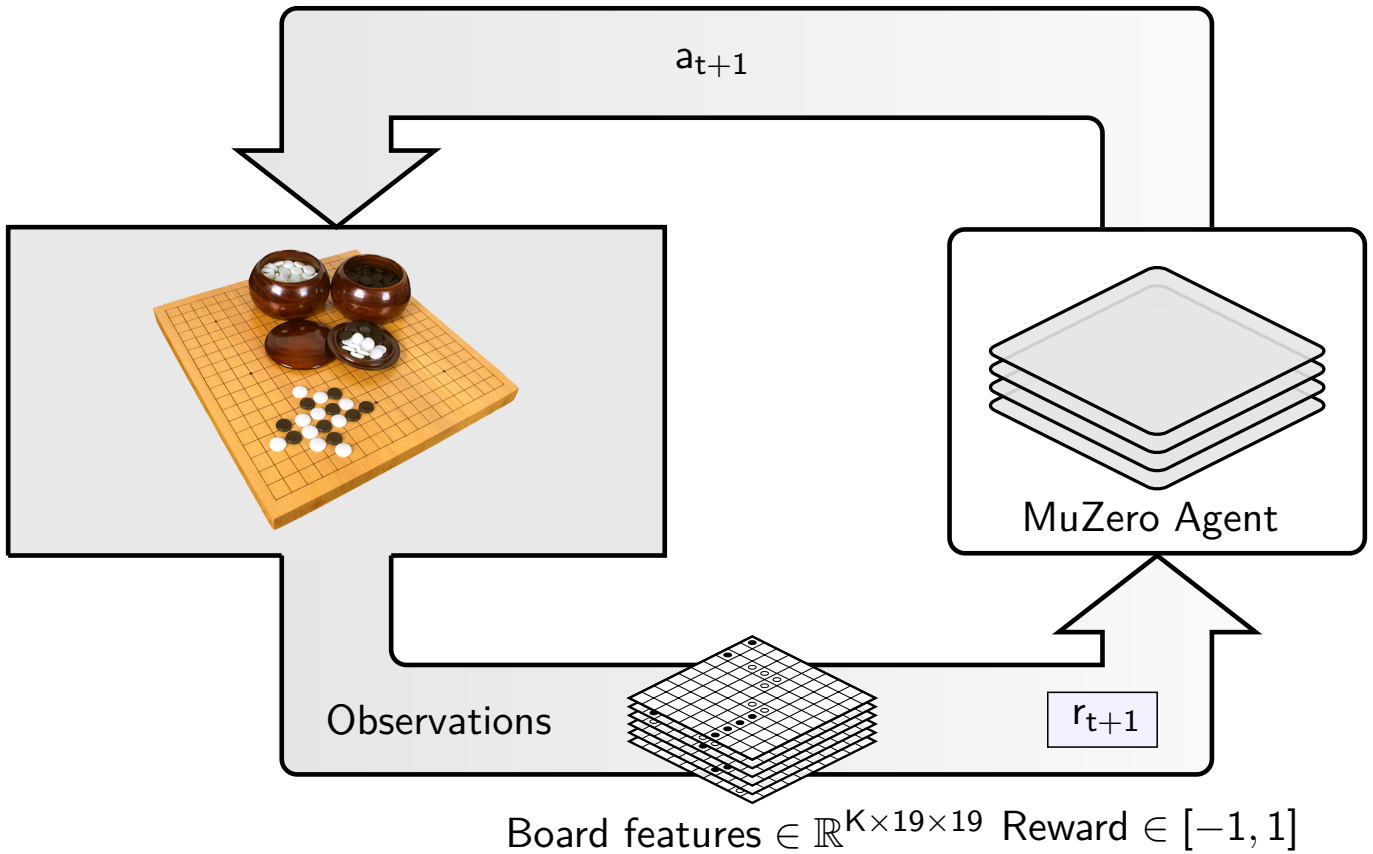
kimham@kth.se

Division of Network and Systems Engineering
KTH Royal Institute of Technology

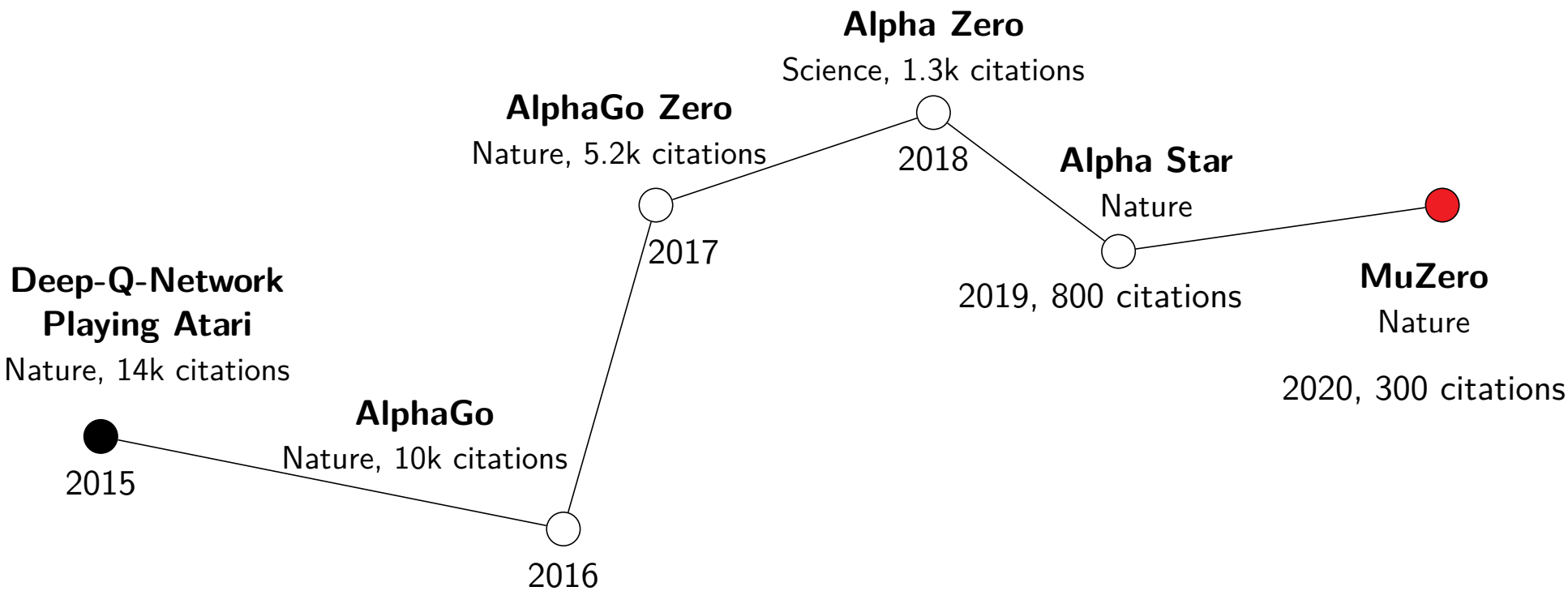
April 23, 2021

The Context and Key Points of the Paper

- MuZero is a **model-based RL Algorithm**
- Target domain: **Games** (Atari, Go, Chess..)
- A **function-approximation** approach to model-learning
- Extension to the **Alpha-Series**: AlphaGo, AlphaZero, ...



The Context and Key Points of the Paper

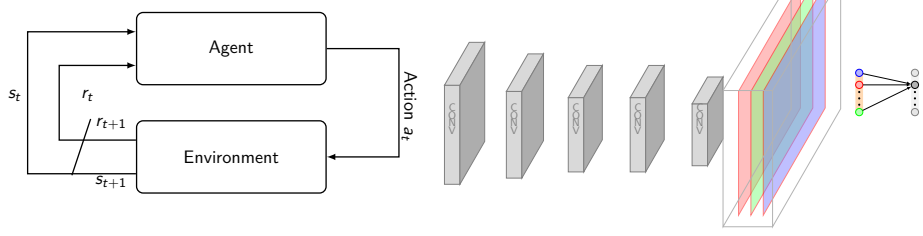


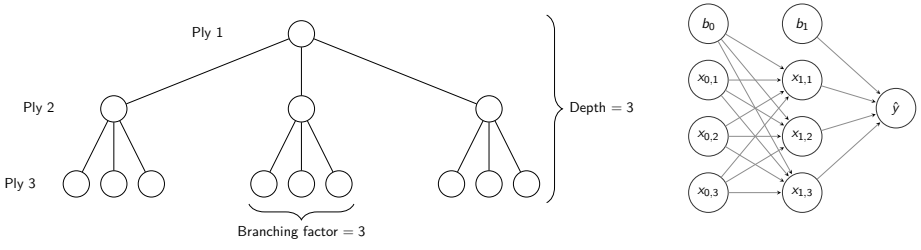
Outline

- **Background**
  - Why Games
  - The Alpha Series
    - AlphaGo
    - AlphaGo Zero
- **The MuZero Algorithm**
- **Limitations**
  - Limitations of MuZero
- **Applications to Security?**
- **Conclusions**

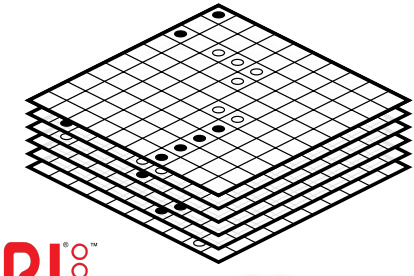
Background: Why Games

AI & Machine Learning

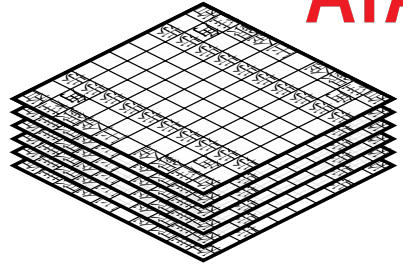


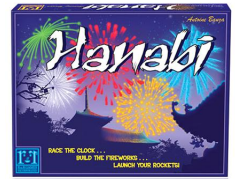


Games



ATARI 2600

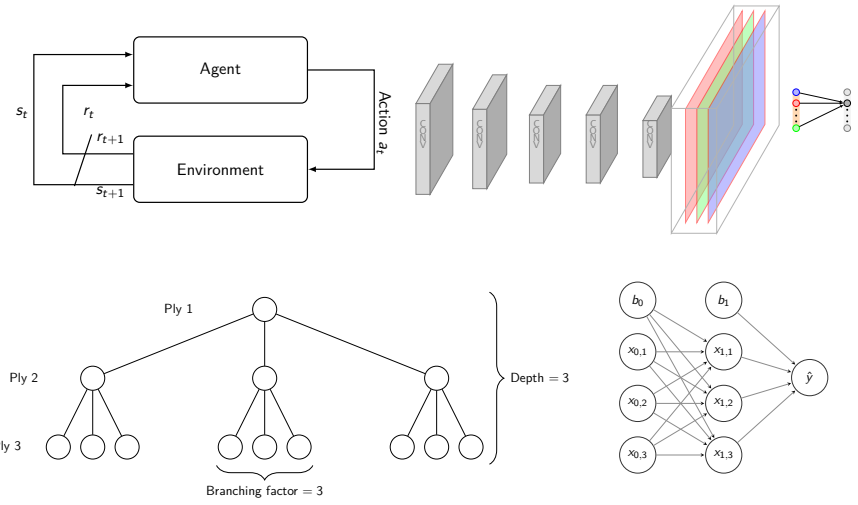




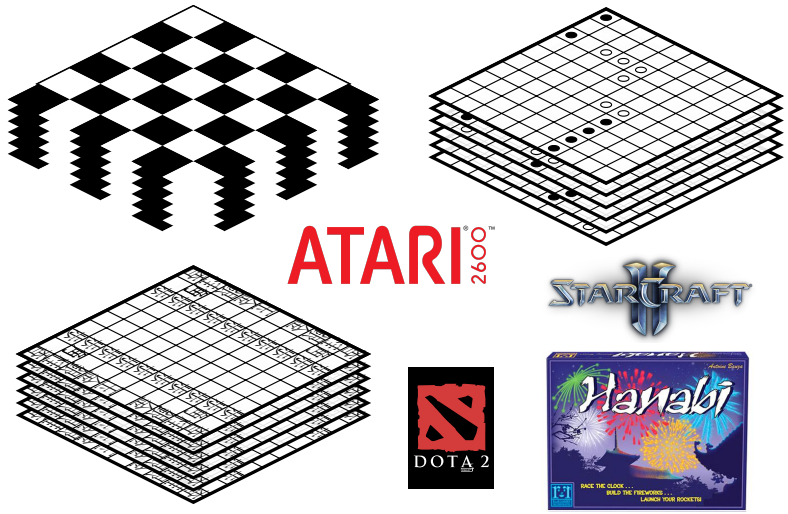
Why Combine the two?

Background: Why Games

AI & Machine Learning



Games



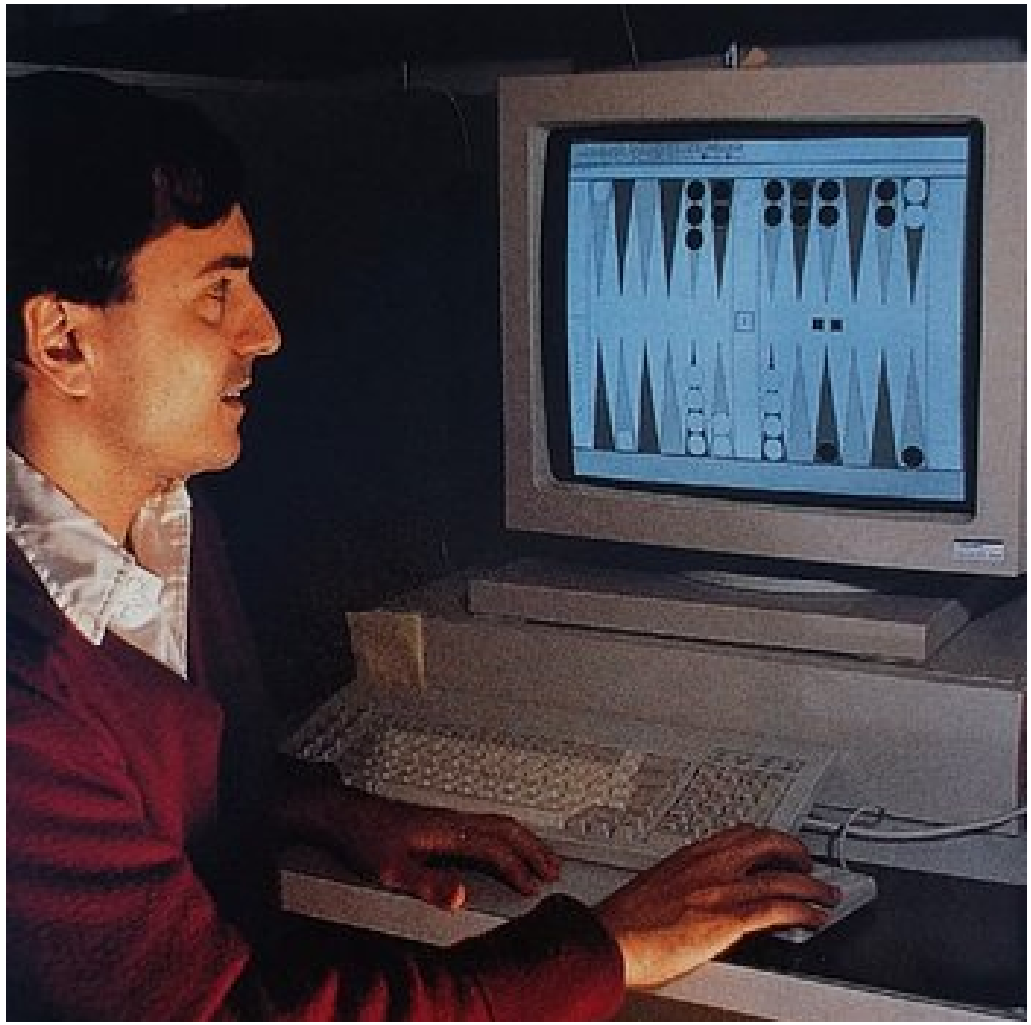
Why Combine the two?

- AI & Games have a long history (Turing '50& Minsky 60')
- Simple to evaluate, reproducible, controllable, quick feedback loop
- Common benchmark for the research community

Background: 1997 DeepBlue[1] vs Kasparov

[1] Murray Campbell, A. Joseph Hoane, and Feng-hsiung Hsu. "Deep Blue". In: *Artif. Intell.* 134.1–2 (Jan. 2002), 57–83. ISSN: 0004-3702. DOI: 10.1016/S0004-3702(01)00129-1. URL: https://doi.org/10.1016/S0004-3702(01)00129-1.

Background: 1992 Tesauro's TD-Gammon[2]



[2] Gerald Tesauro. "TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play". In: *Neural Comput.* 6.2 (Mar. 1994), 215–219. ISSN: 0899-7667. DOI: 10.1162/neco.1994.6.2.215. URL: https://doi.org/10.1162/neco.1994.6.2.215.

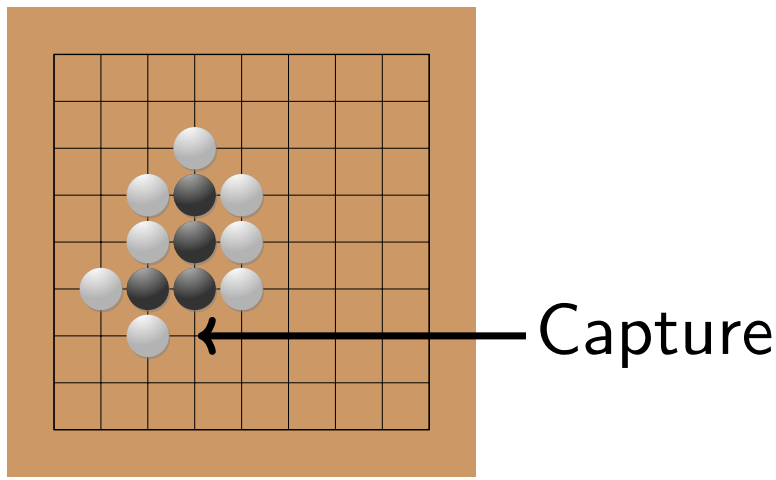
Background: 1959 Arthur Samuel's Checkers Player[3]

[3] A. L. Samuel. "Some Studies in Machine Learning Using the Game of Checkers". In: *IBM J. Res. Dev.* 3.3 (July 1959), 210–229. ISSN: 0018-8646. DOI: 10.1147/rd.33.0210. URL: https://doi.org/10.1147/rd.33.0210, A. L. Samuel. "Some Studies in Machine Learning Using the Game of Checkers". In: *IBM J. Res. Dev.* 3.3 (July 1959), 210–229. ISSN: 0018-8646. DOI: 10.1147/rd.33.0210. URL: https://doi.org/10.1147/rd.33.0210.

AlphaGo

- AlphaGo is a **system** that combines:
  - Imitation learning (behavioral cloning)
  - Reinforcement learning (self-play)
  - Planning (look-ahead search with MCTS)
  - Massive computational power (Google's data center)

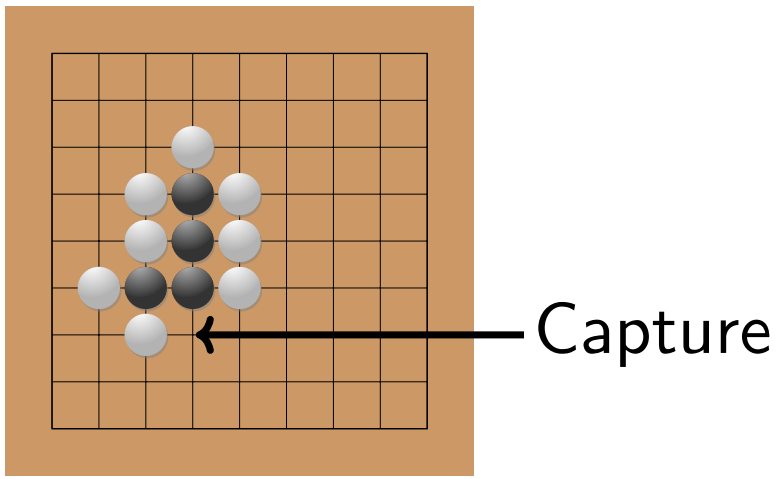
- AlphaGo beat Lee Sedol (WC in Go) in 2016
- AlphaGo **assumes access to a simulation model** $(s_t, a_t) \rightarrow s_{t+1}$ that can be used for look-ahead search
  - *This assumption will be relaxed in MuZero later...*
- **Key Idea**: Guided & truncated look-ahead search using neural network predictions.



AlphaGo

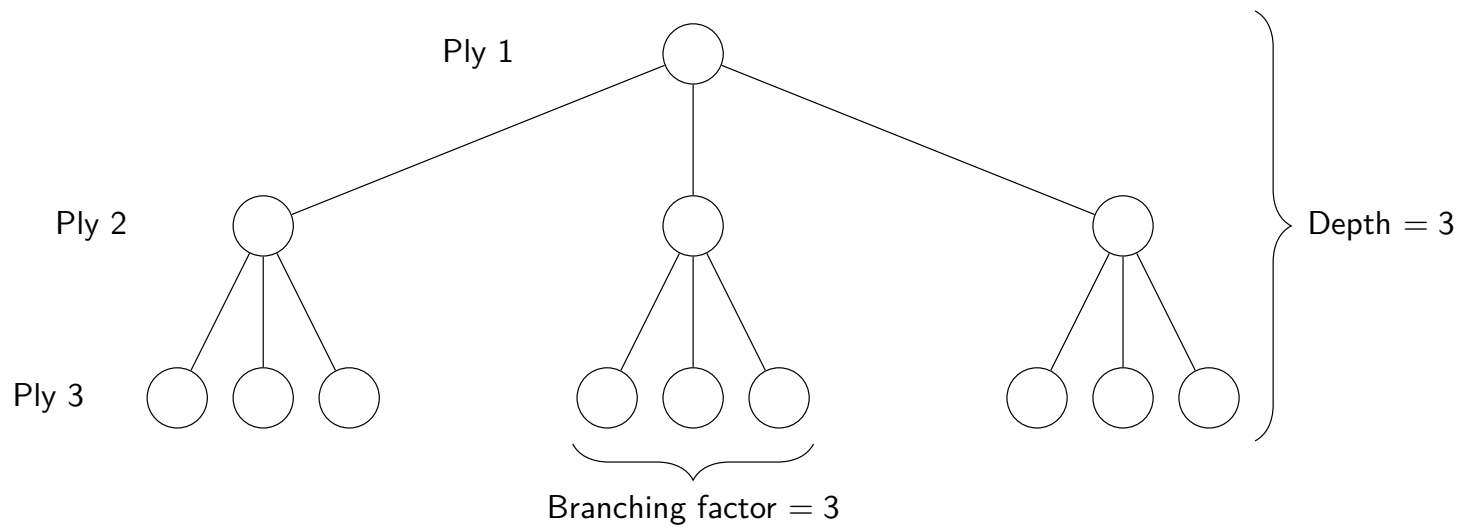
- AlphaGo is a **system** that combines:
  - Imitation learning (behavioral cloning)
  - Reinforcement learning (self-play)
  - Planning (look-ahead search with MCTS)
  - Massive computational power (Google's data center)

- AlphaGo beat Lee Sedol (WC in Go) in 2016
- AlphaGo **assumes access to a simulation model** $(s_t, a_t) \rightarrow s_{t+1}$ that can be used for look-ahead search
  - *This assumption will be relaxed in MuZero later...*
- **Key Idea**: Guided & truncated look-ahead search using neural network predictions.

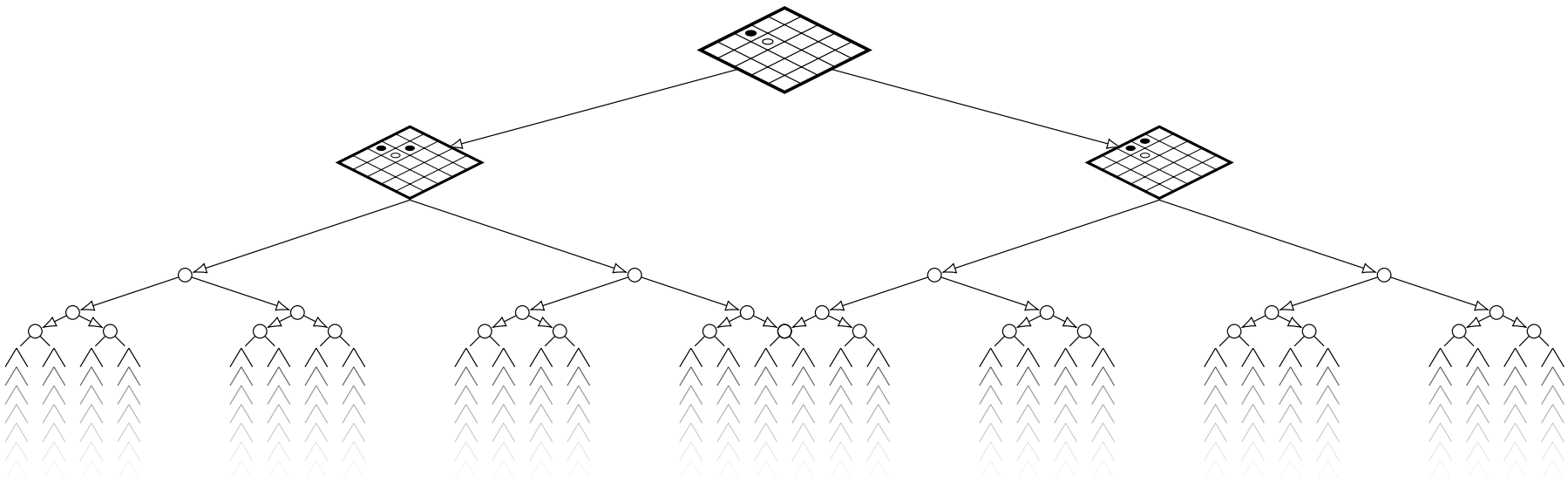


Game Trees

- How do you program a computer to play a board game?
- Simplest approach:
  - (1) Program a game tree; (2) Assume opponent think like you; (3) Look-ahead and evaluate each move
  - **Requires Knowledge of game rules and evaluation function**

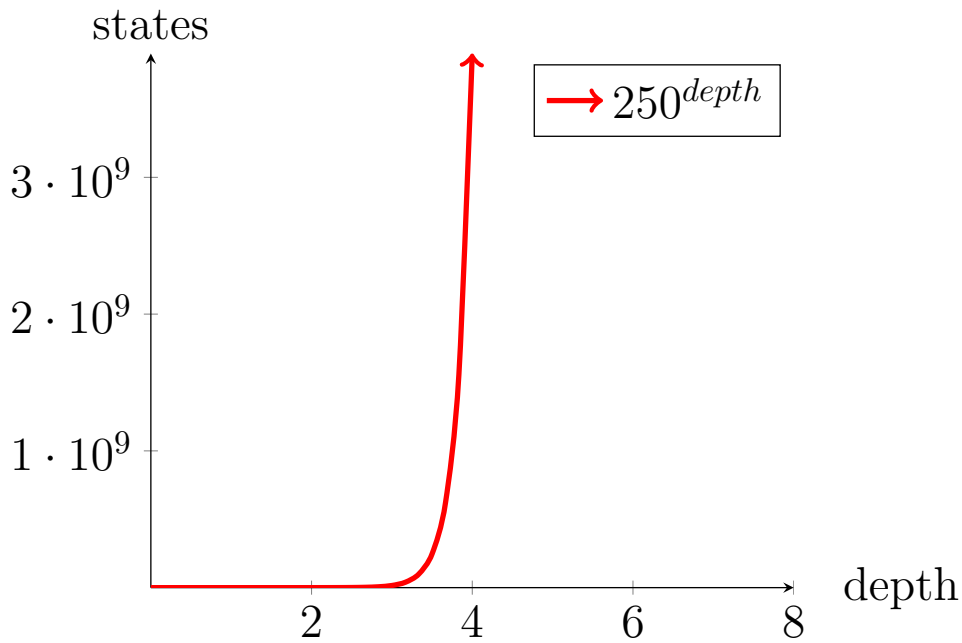


Search + Go = 💣



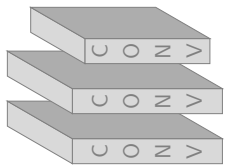
Some Numbers

- **Atoms in the universe**
  - $\approx 10^{80}$
- **States**
  - Go: 10^{170}, Chess: 10^{47}
- **Game tree complexity**
  - Go: 10^{360}, Chess: 10^{123}
- **Average branching factor**
  - Go: 250, Chess: 35
- **Board size (positions)**
  - Go: 361, Chess: 64



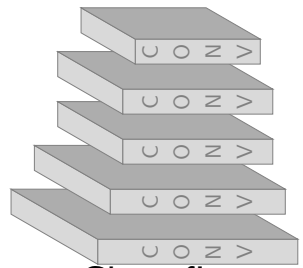
AlphaGo Training Pipeline (1/2)

Supervised Rollout Policy Network
$p_\pi(a|s)$

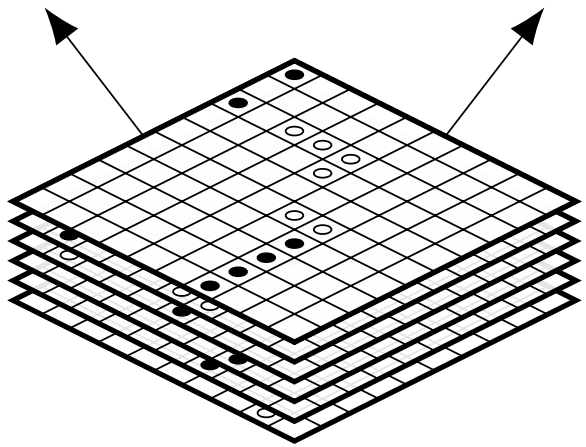


Classification
$\min_{p_\pi} \mathcal{L}(\text{predicted move}, \text{expert move})$

Supervised Policy Network
$p_\sigma(a|s)$



Classification
$\min_{p_\sigma} \mathcal{L}(\text{predicted move}, \text{expert move})$

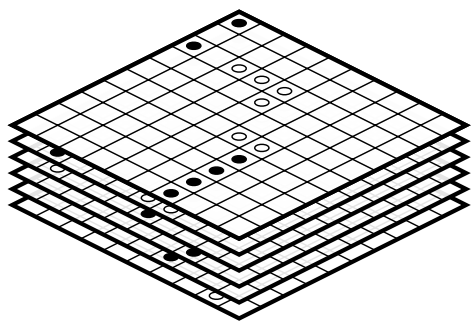


Human Expert Moves $\mathcal{D}_1$

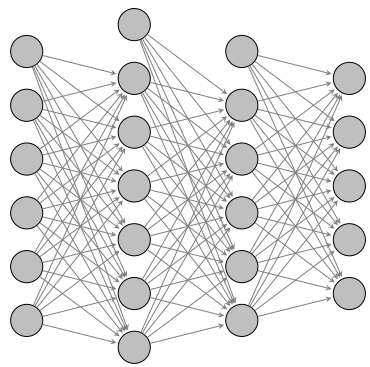
AlphaGo Training Pipeline (1/2): Imitation Learning

Expert Demonstrations

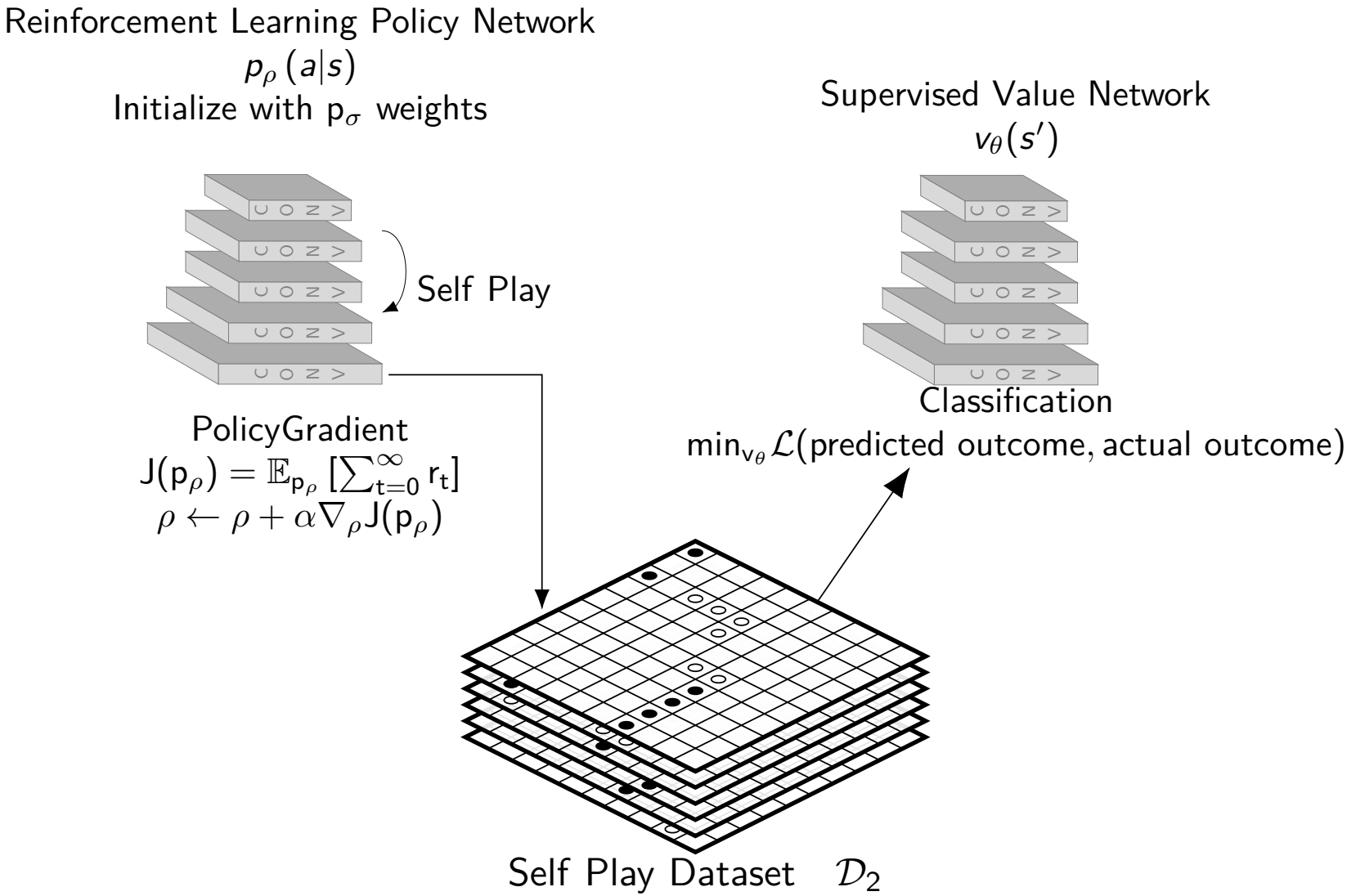
$\langle$State, Action$\rangle$ pairs



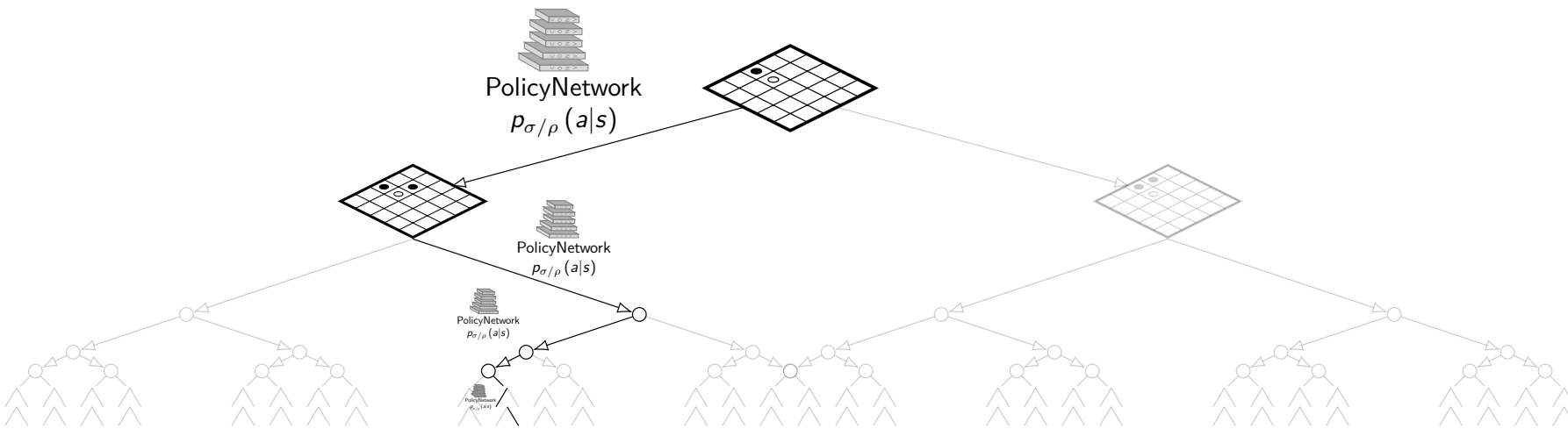
Supervised Learning



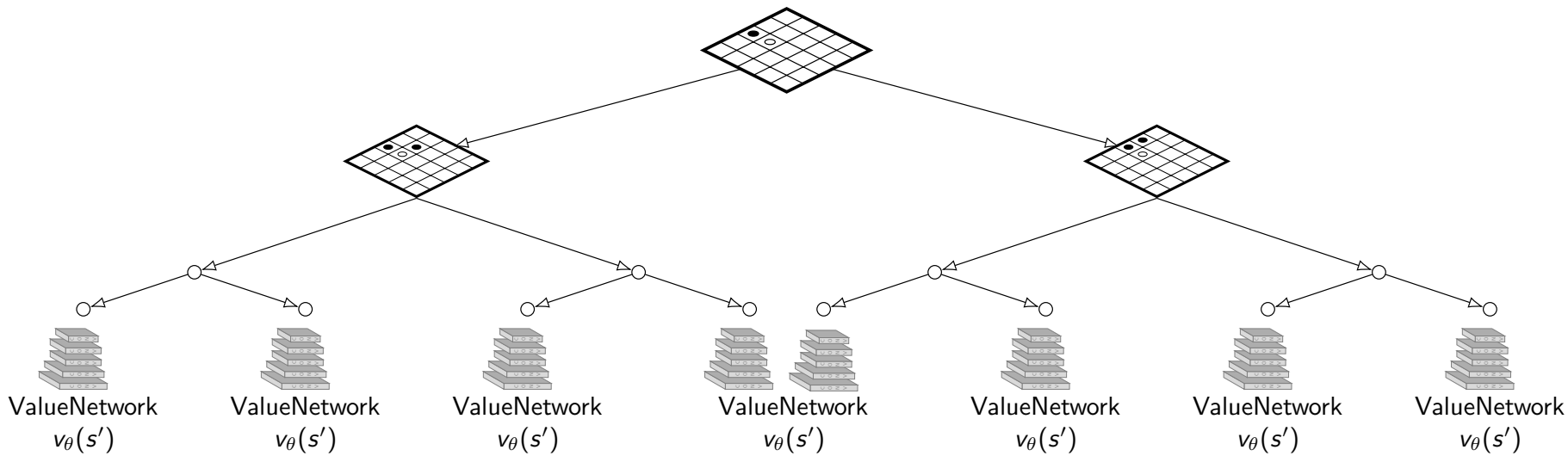
AlphaGo Training Pipeline (2/2)



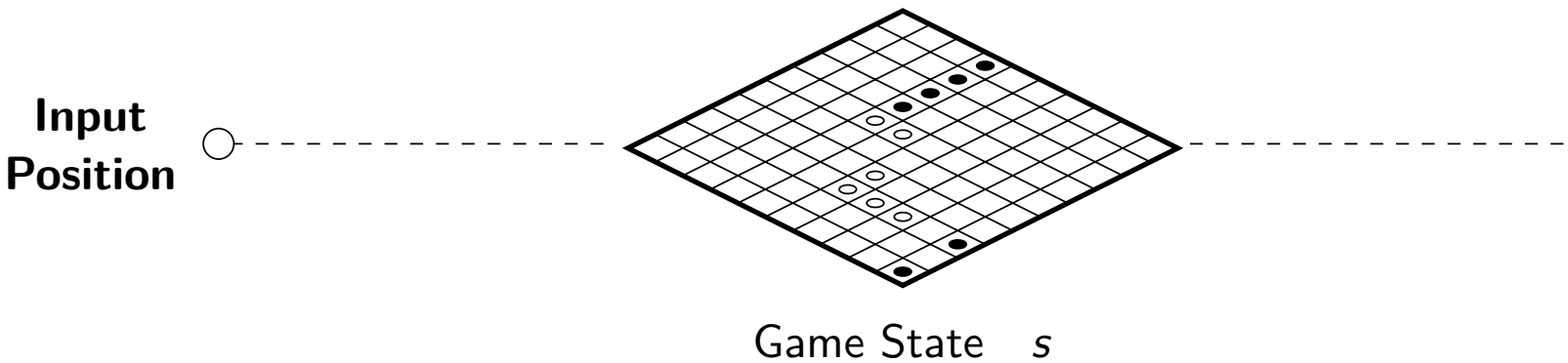
Guided Search Search Using the Policy Network



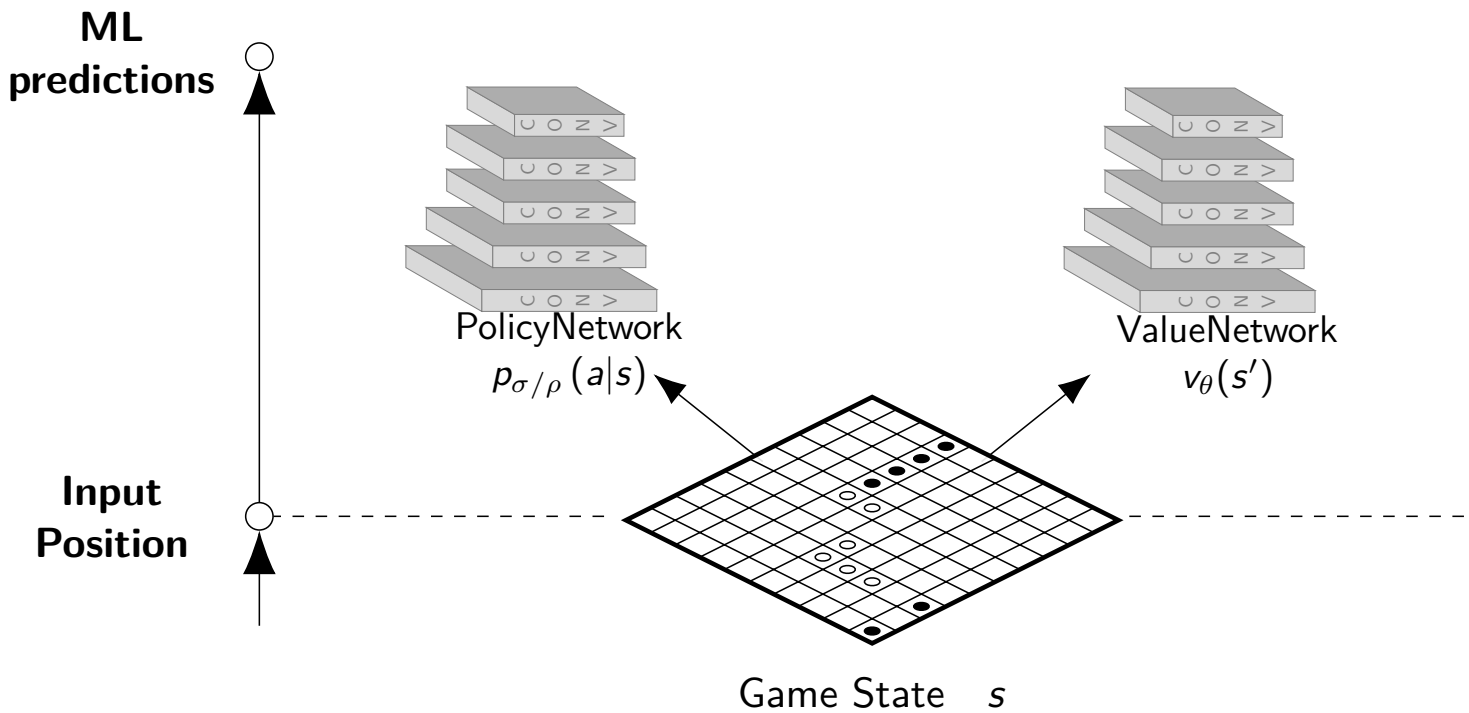
Depth-Limited Search Using the Value Network



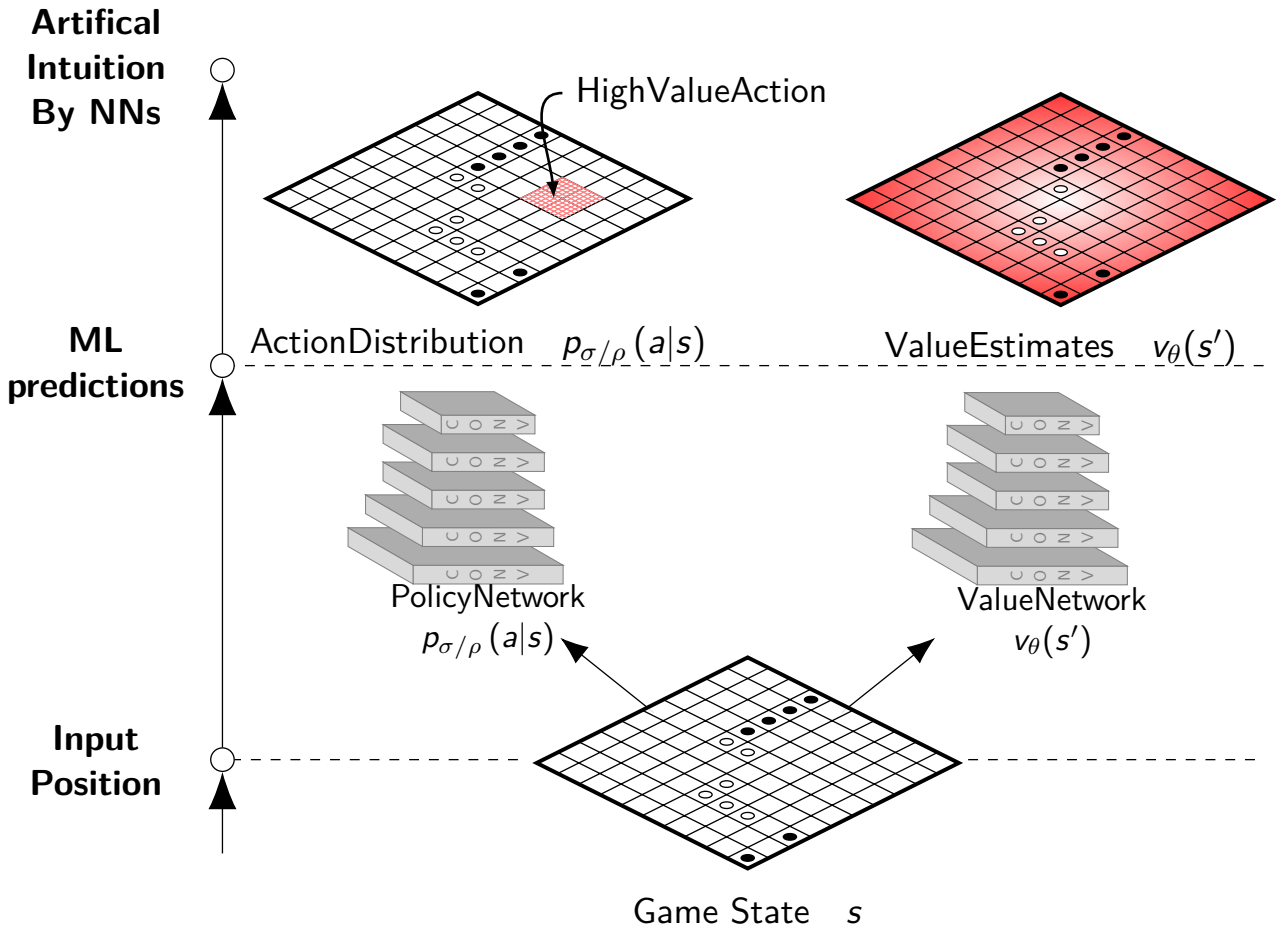
AlphaGo Prediction Pipeline



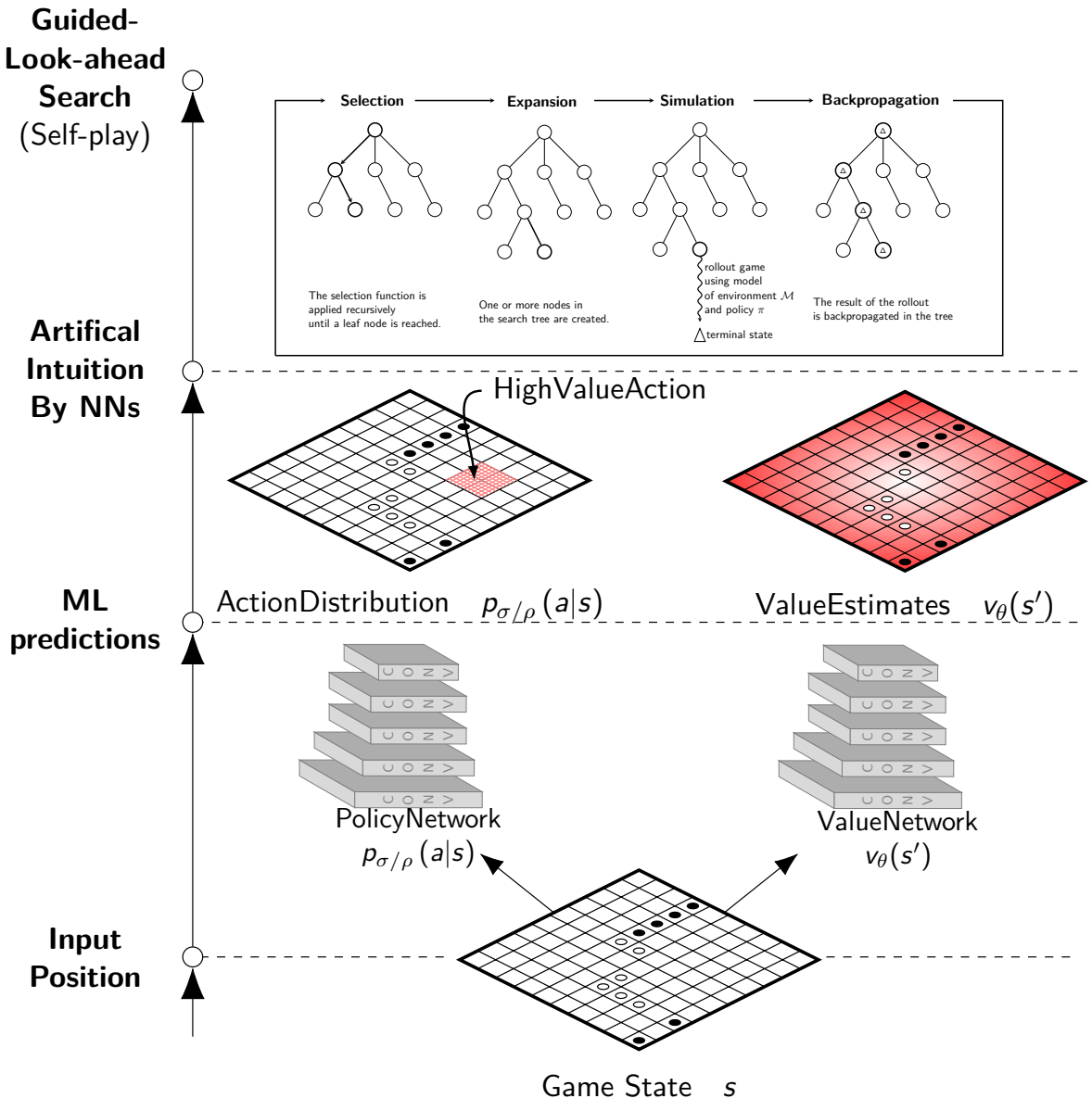
AlphaGo Prediction Pipeline



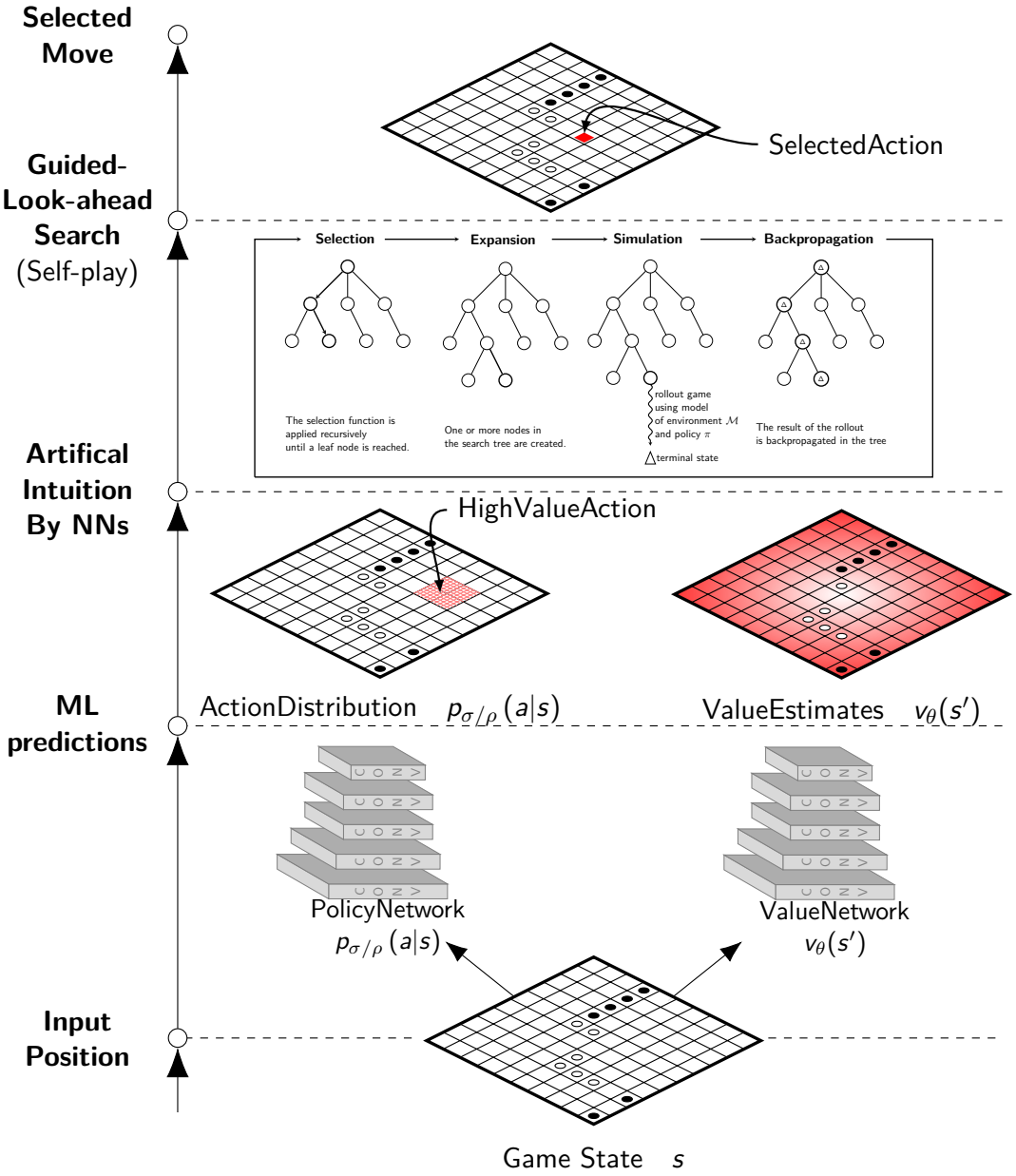
AlphaGo Prediction Pipeline



AlphaGo Prediction Pipeline



AlphaGo Prediction Pipeline



AlphaZero

- ~~AlphaGo~~ AlphaZero is a **system** that combines:
  - ~~Imitation learning (behavioral cloning)~~
  - Reinforcement learning (self-play)
  - Planning (look-ahead search with MCTS)
  - Massive computational power (Google's data center)

- AlphaZero works on Chess, Go, Shogi

- ~~AlphaGo~~ AlphaZero **assumes access to a simulation model** $(s_t, a_t) \to s_{t+1}$ that can be used for look-ahead search
  - *This assumption will be relaxed in MuZero later...*

- **Key Idea**: Guided & truncated look-ahead search using neural network predictions.

AlphaZero

- ~~AlphaGo~~ AlphaZero is a **system** that combines:
  - ~~Imitation learning (behavioral cloning)~~
  - Reinforcement learning (self-play)
  - Planning (look-ahead search with MCTS)
  - Massive computational power (Google's data center)

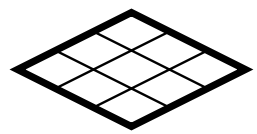
- AlphaZero works on Chess, Go, Shogi

- ~~AlphaGo~~ AlphaZero **assumes access to a simulation model** $(s_t, a_t) \rightarrow s_{t+1}$ that can be used for look-ahead search
  - *This assumption will be relaxed in MuZero later...*

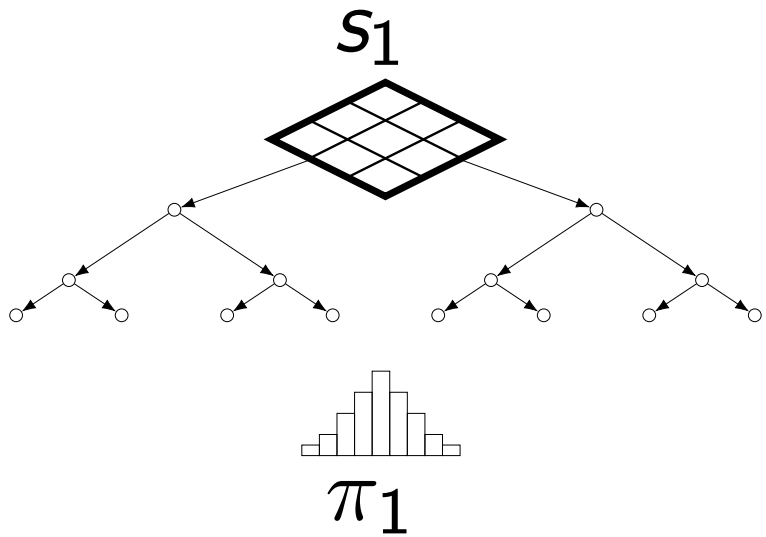
- **Key Idea**: Guided & truncated look-ahead search using neural network predictions.

AlphaZero Self-Play Training Algorithm

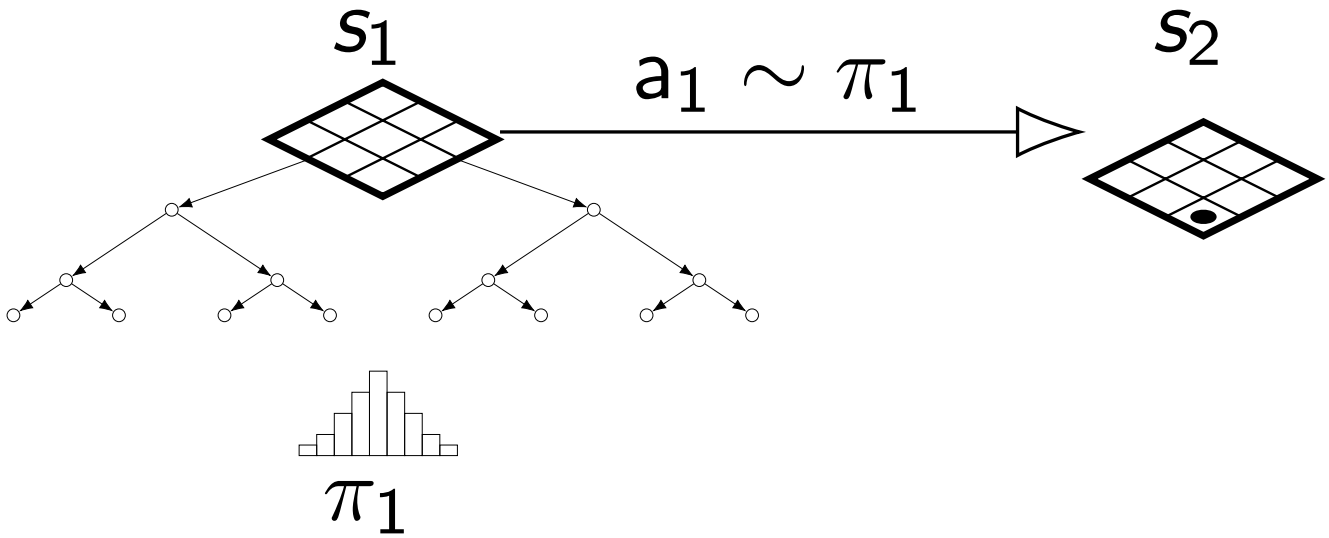
s_1



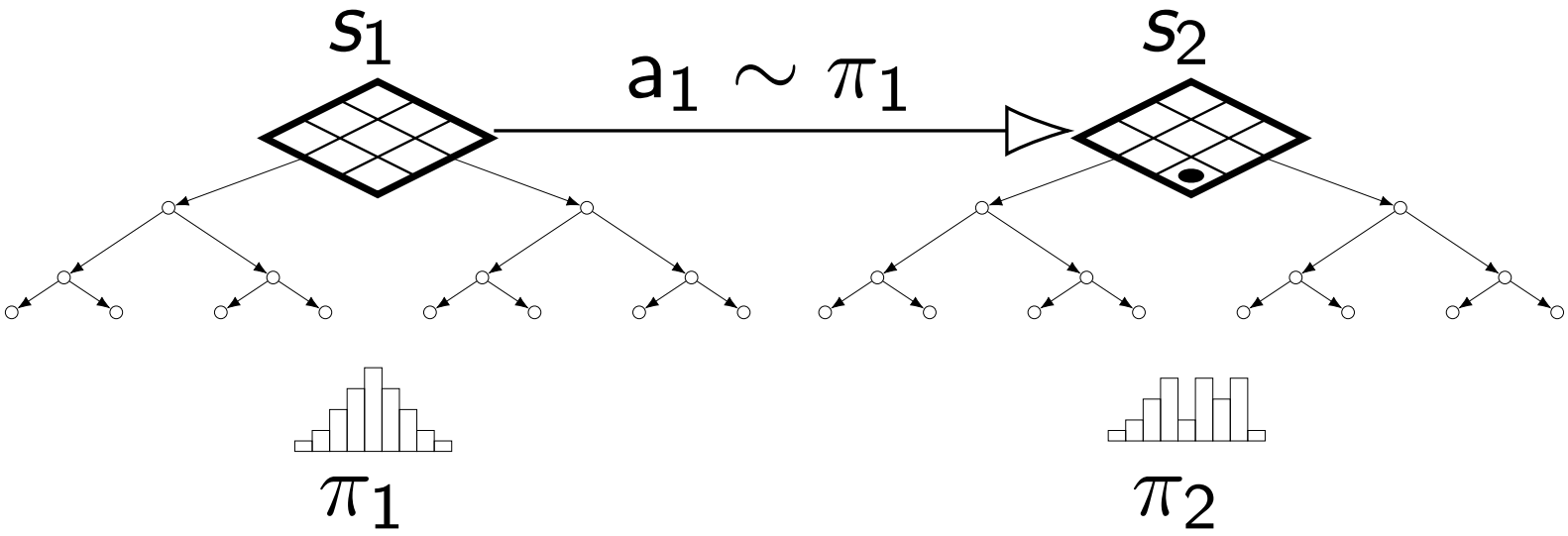
AlphaZero Self-Play Training Algorithm



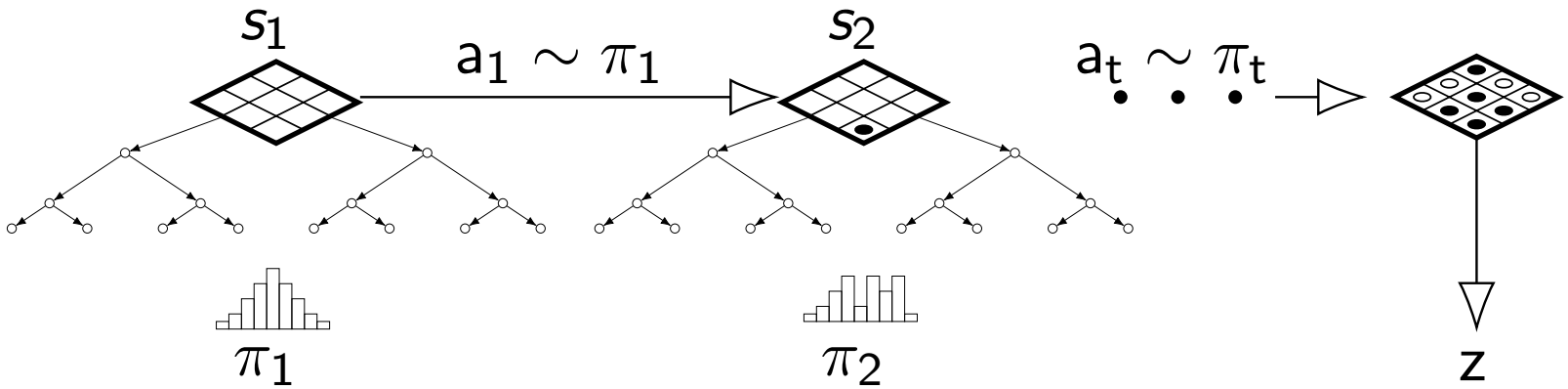
AlphaZero Self-Play Training Algorithm



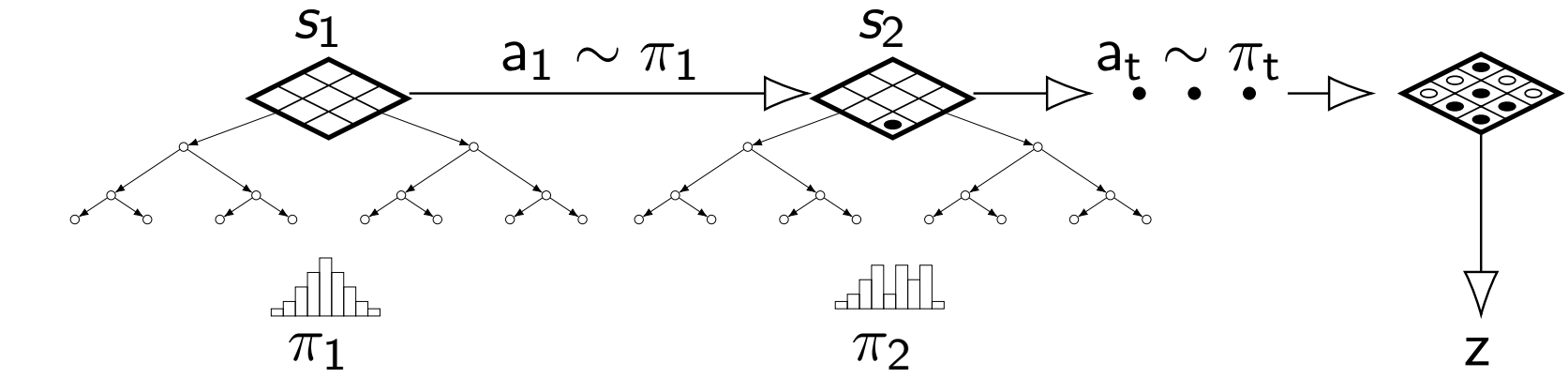
AlphaZero Self-Play Training Algorithm



AlphaZero Self-Play Training Algorithm



AlphaZero Self-Play Training Algorithm

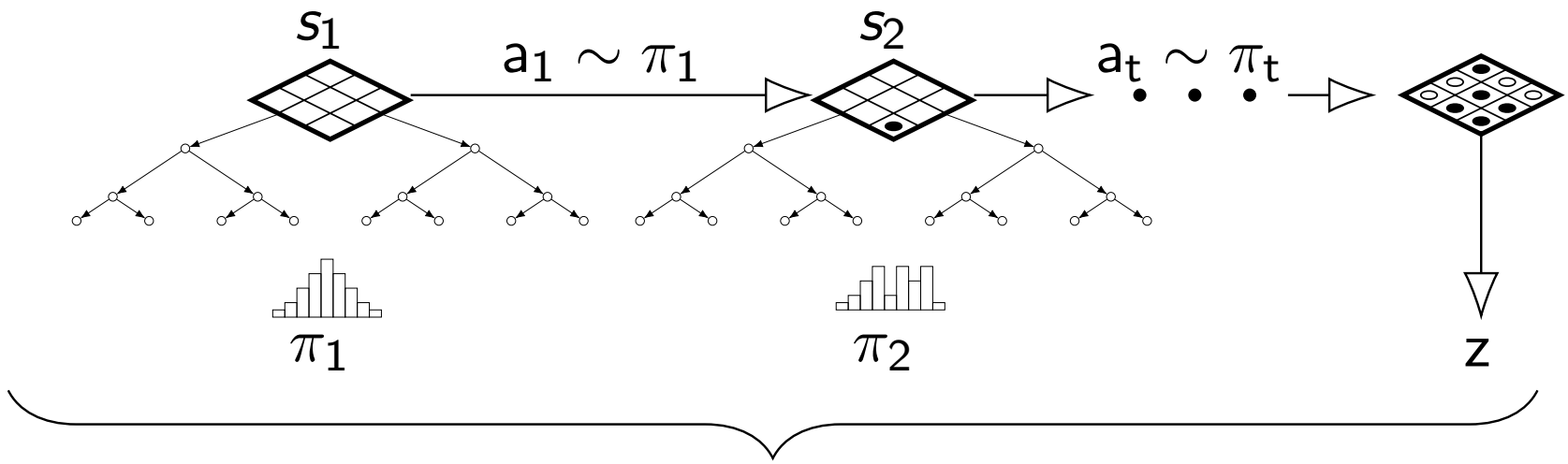


Self Play Training Data

$f_\theta(s_t) = (p_t, v_t)$

$\theta' = \theta - \alpha \nabla_\theta \mathcal{L}((p_t, v_t), (\pi_t, z))$

AlphaZero Self-Play Training Algorithm

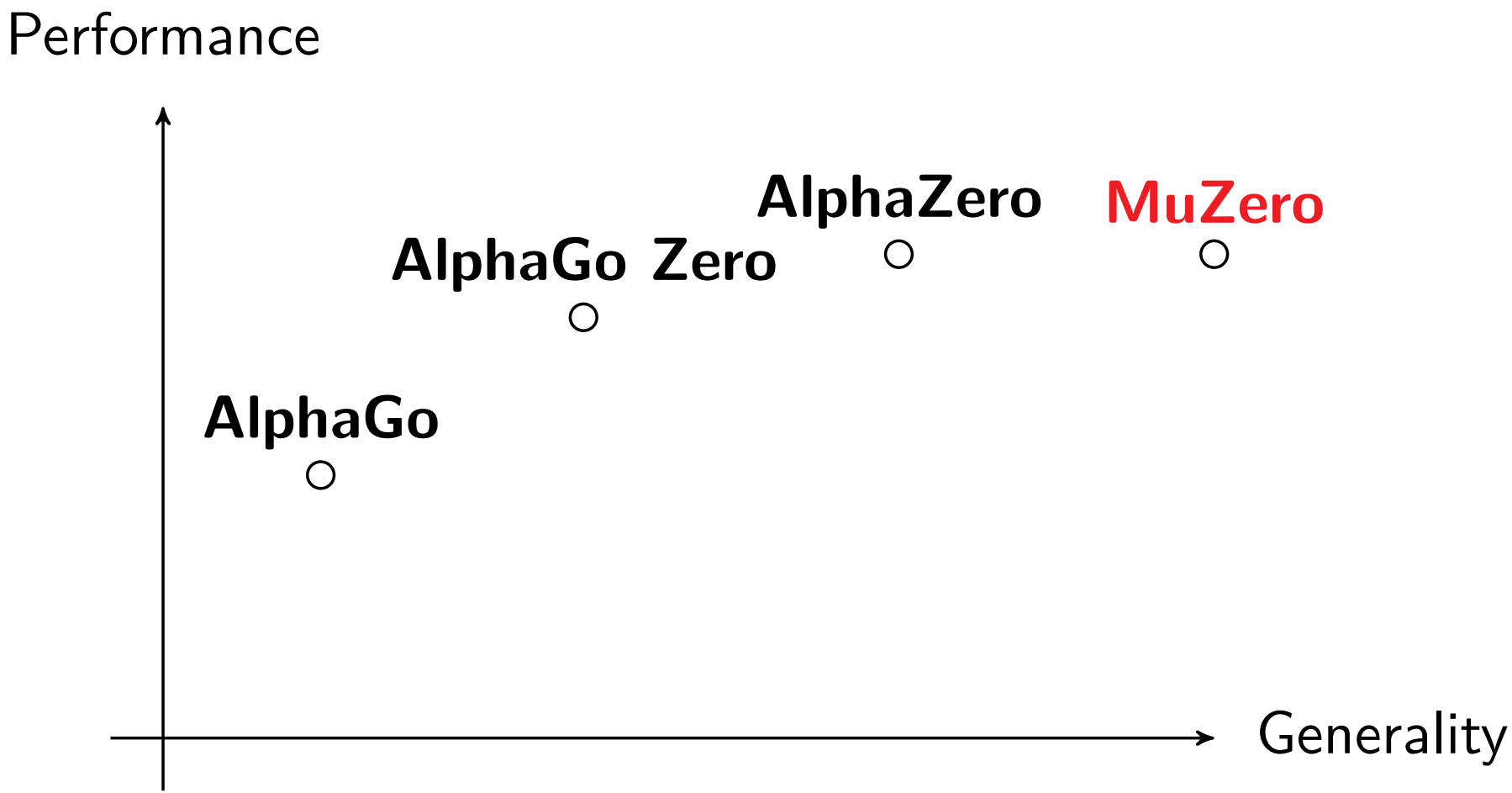


Self Play Training Data

$$f_\theta(s_t) = (p_t, v_t) \qquad \theta' = \theta - \alpha \nabla_\theta \mathcal{L}((p_t, v_t), (\pi_t, z))$$

$$\mathcal{L}(f_\theta(s_t), (\pi_t, z)) = \underbrace{(z - v_t)^2}_{\text{MSE}} - \underbrace{\pi_t^T \log p_t + c||\theta||^2}_{\text{Cross-entropy loss}}$$

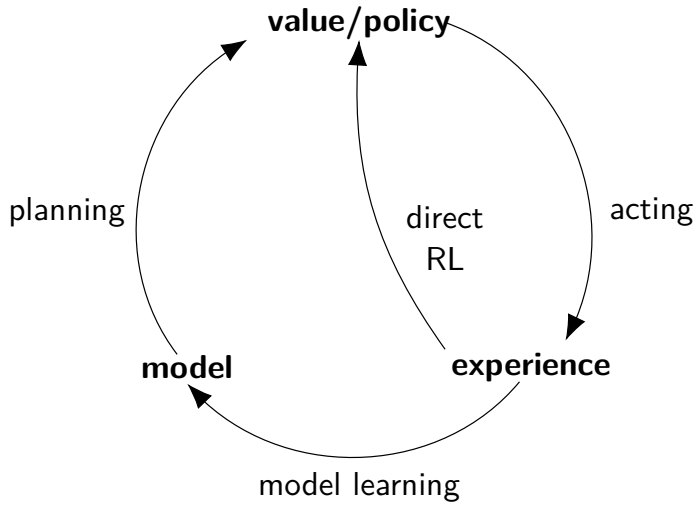
From Specialized to General



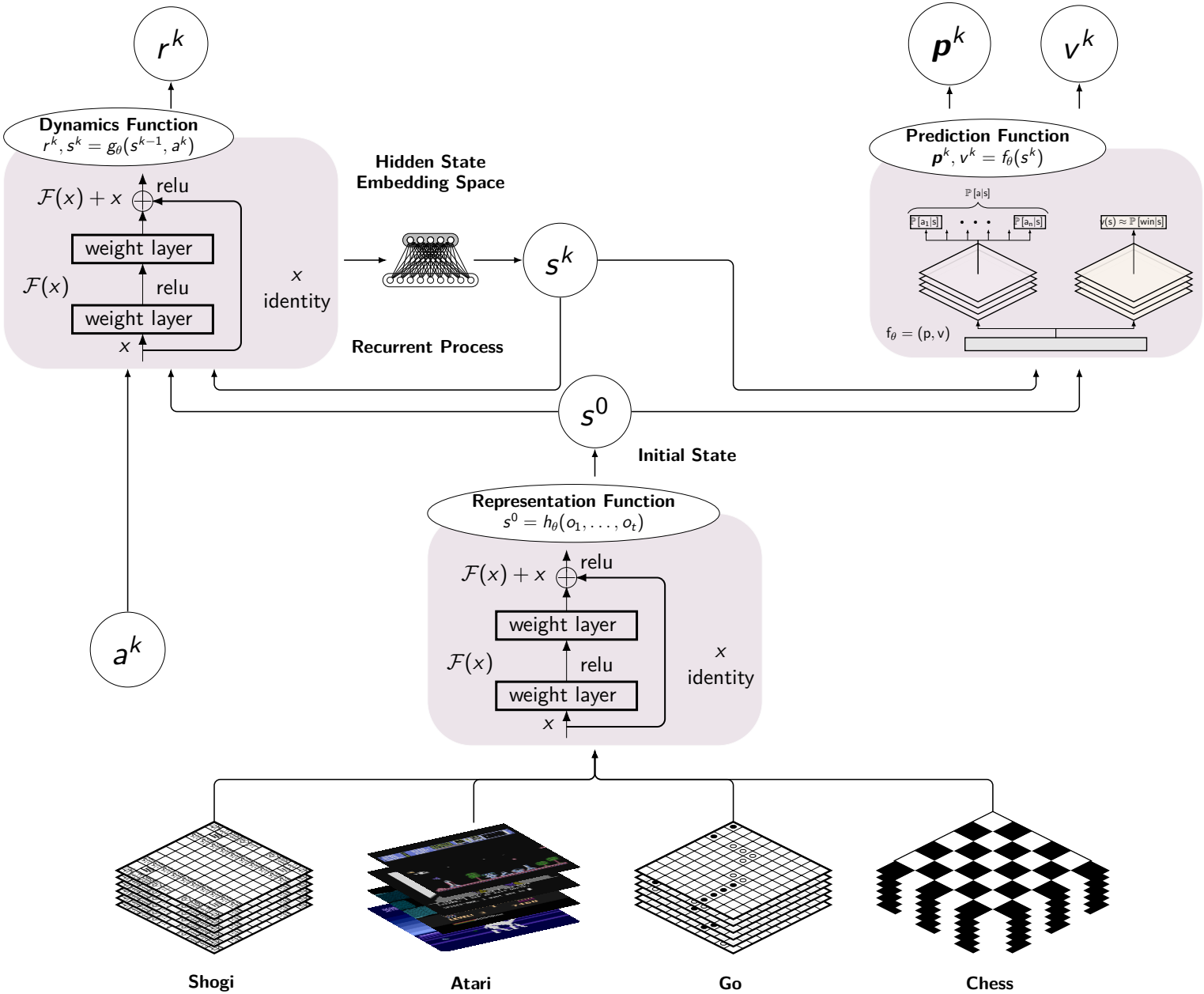
The MuZero Algorithm

- AlphaGo, AlphaZero **assume access to a simulation model**
- MuZero does **not** assume access to a simulation model

- MuZero is a model-based RL algorithm that **learns** a model
- MuZero differs from traditinal model-based RL algorithms by:
  1. The algorithm is based on **function approximation**
  2. The algorithm learns the model **implicitly**
  3. The algorithm **uses look-ahead search in a novel way**
  4. Specialized **neural architecture**



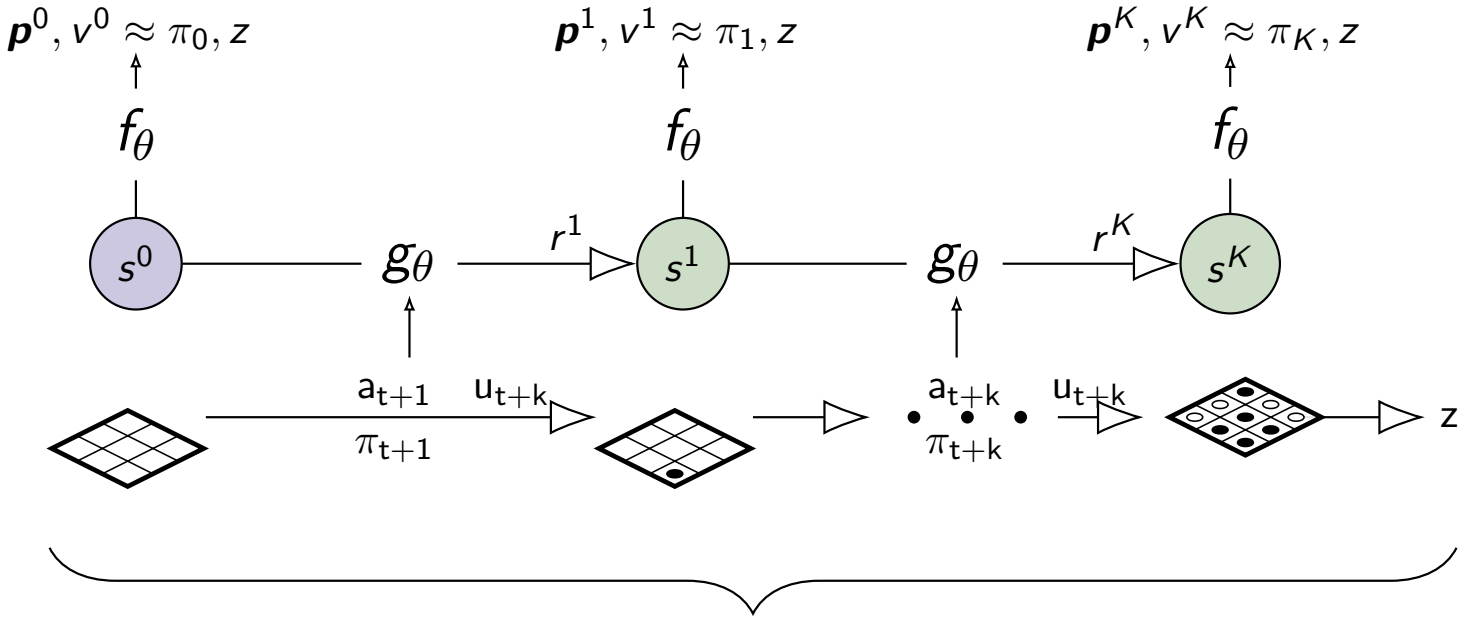
The MuZero Algorithm: Neural Network



The MuZero Algorithm: Training

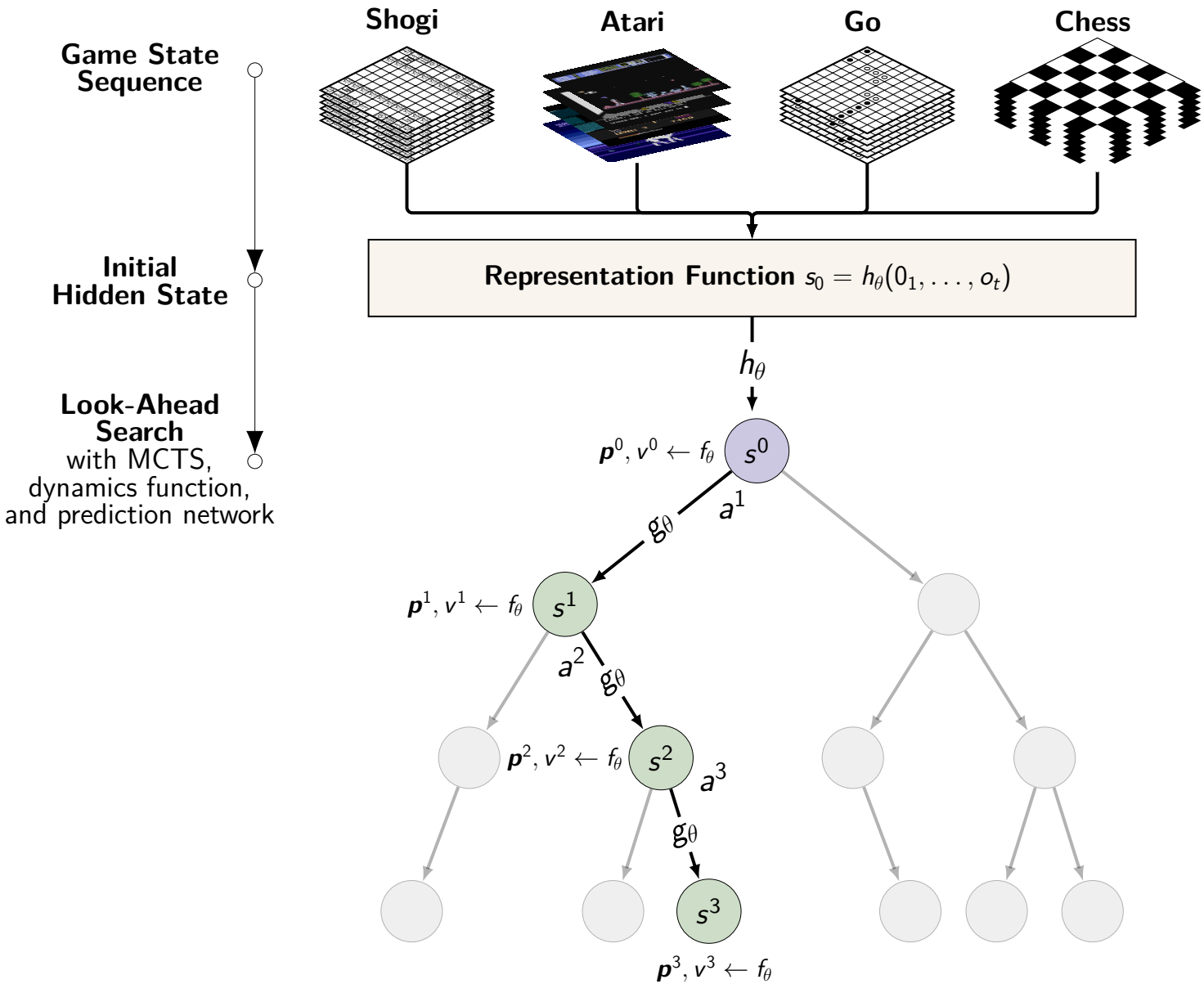
Loss: $l_t\theta = \sum_{k=0}^{K} l^r(u_{t+k}, r_{t_t^k}) + l^v(z_{t+k}, v_t^k) + l^p(\pi_{t+k}, p_t^k) + c\theta^t$

Lookahead search using Model of Environment



Sample Trajectory From Environment

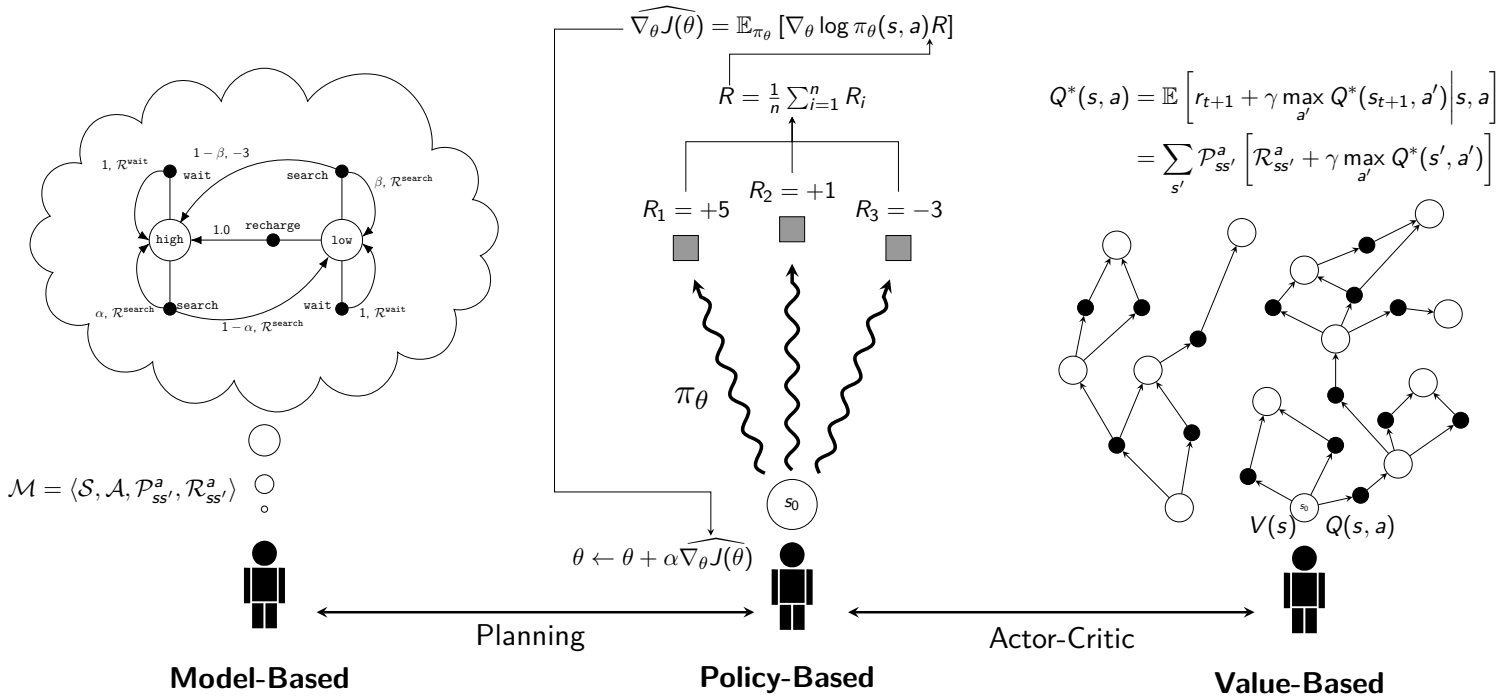
The MuZero Algorithm: Planning



Limitations of MuZero and Drawbacks of the Paper

1. **Sample efficiency**:
   - MuZero reduces need for prior knowledge by learning a model
   - However, MuZero is not sample efficient

2. **Cannot learn stochastic models**:
   - This is left for future work
   - Many practical models are stochastic



Applications to Security?

- **Many infrastructures cannot be described by simple equations**
  - Alternative approach: estimate/learn a model
- Benefit of **model-based RL**: more sample efficient (generally)
- **Can we use MuZero to Learn the Model?**
  - In principle, yes.
  - In practice, no.
    - Muzero is likely to be too sample inefficient for realistic infrastructures

Conclusions

- MuZero is a **new model-based RL algorithm**
- MuZero **generalizes AlphaZero** to not require prior knowledge about the environment dynamics
- MuZero uses **implicit learning of environment dynamics**
- MuZero is a **new state of the art** on several games
- **Can we use MuZero for other domains than games?**
  - MuZero reduces need for prior knowledge but does not reduce the number of samples required
  - MuZero has some interesting ideas, but needs to be developed further (IMO)