~~Active Learning~~ Self-Play in the age of Deep Learning

Wouter M. Koolen

Centrum Wiskunde & Informatica

YES X

EURANDOM, Tuesday 19th March, 2019

Outline

Main Question

Beyond supervised learning

What data should the learning system collect?

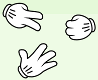
Active learning: query labels of instances. Learn good classifier.
Reinforcement learning: obtain rewards. Learn good policy.

This talk

Let's look at deep neural networks in games.

- Learning system decides which data are collected.
- Here: self-play.

Matrix game



Consider a two-player finite zero sum game.

M =

1	4
3	2

Goal: find Nash Equilibrium in mixed strategies (saddle point) witnessing value.

$$V^* \;=\; \min_{p \in \triangle} \max_{q \in \triangle} p^\intercal M q$$

Matrix game

Q: how to solve a matrix game?

Matrix game

Q: how to solve a matrix game?
A: LP

Matrix game

Q: how to solve a matrix game?
A: LP

Theorem (Freund and Schapire 1999)

Instantiate a **no-regret** *learning algorithm for each player. Then the average iterates converge to Nash equilibrium.*

Matrix game

Q: how to solve a matrix game?
A: LP

Theorem (Freund and Schapire 1999)

Instantiate a **no-regret** *learning algorithm for each player. Then the average iterates converge to Nash equilibrium.*

Many many refinements, e.g. [Rakhlin and Sridharan, 2013].

Matrix game

Q: how to solve a matrix game?
A: LP

Theorem (Freund and Schapire 1999)

Instantiate a **no-regret** *learning algorithm for each player.*
Then the average iterates converge to Nash equilibrium.

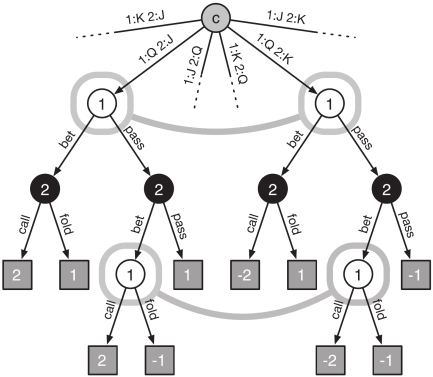
Many many refinements, e.g. [Rakhlin and Sridharan, 2013].

Q: what if second player **doing something else** (e.g. human)?

Outline

Poker

Partial information game.
Heads up limit hold'em version of Poker.



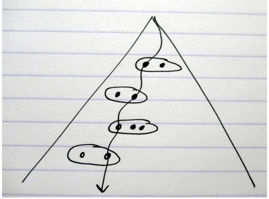
Bowling et al. [2015].

Solution [Bowling et al., 2015]

3.16×10^{17} states.
1.38×10^{13} information sets (decision points).

Cepheus: Instantiate a no-regret learner for each information set.

Interact the hell out of it (900 core-years).

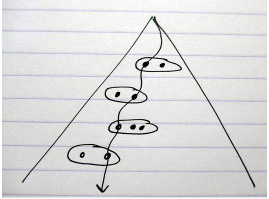


Solution [Bowling et al., 2015]

3.16×10^{17} states.
1.38×10^{13} information sets (decision points).

Cepheus: Instantiate a no-regret learner for each information set.

Interact the hell out of it (900 core-years).



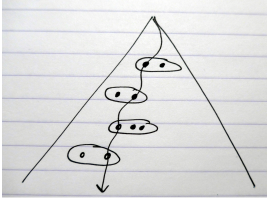
Able to compute sub-optimality of pair of strategies. ≈ 0.

Solution [Bowling et al., 2015]

3.16×10^{17} states.
1.38×10^{13} information sets (decision points).

Cepheus: Instantiate a no-regret learner for each information set.

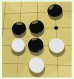
Interact the hell out of it (900 core-years).



Able to compute sub-optimality of pair of strategies. ≈ 0.

Many details (counterfactual regret minimisation, regret matching+, compression, no iterate averaging, . . .)

Go



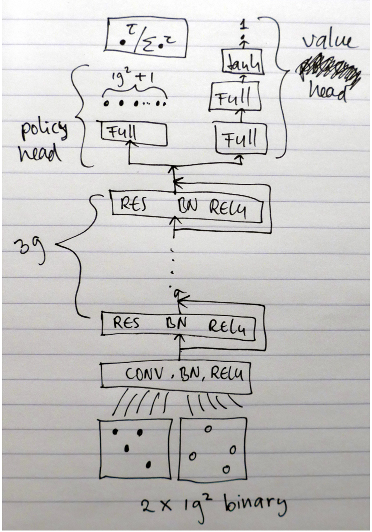
Full information.
Number of states: 10^{170} (Wikipedia)

Need to approximate / generalise.

AlphaGo Zero Birds eye view

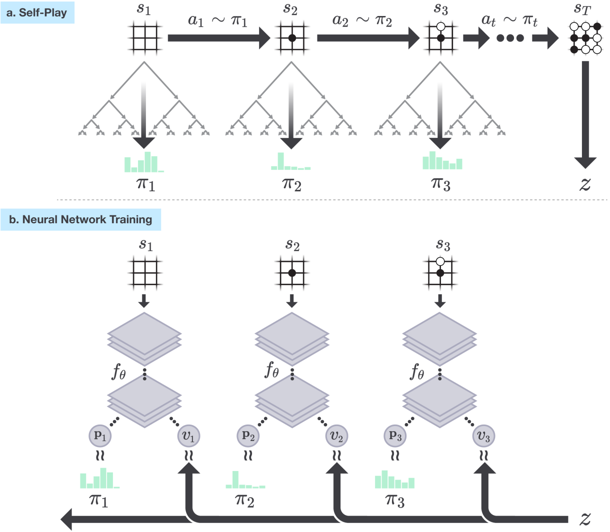
- Starting from zero.
- Neural network f_θ
  - Input: board configuration
  - Output: move probabilities **and** value.
- Residual blocks of convolutional layers
- ReLU, batch normalisation.

AlphaGo Zero Network

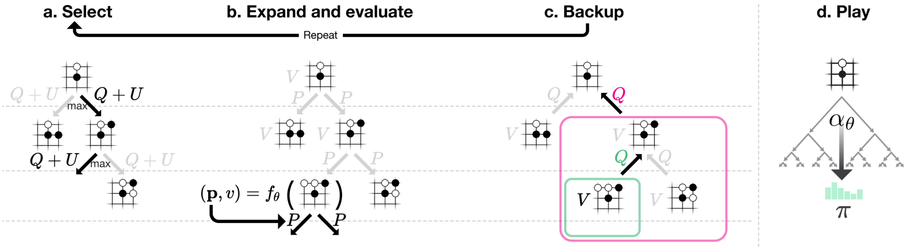


$$f_\theta(s) = (p, v)$$

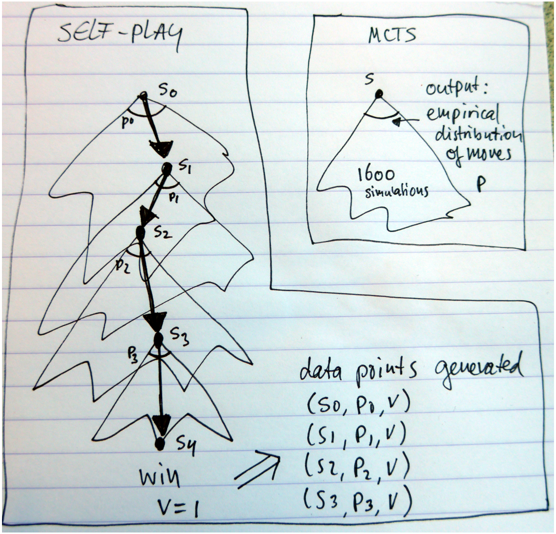
AlphaGo Zero Training



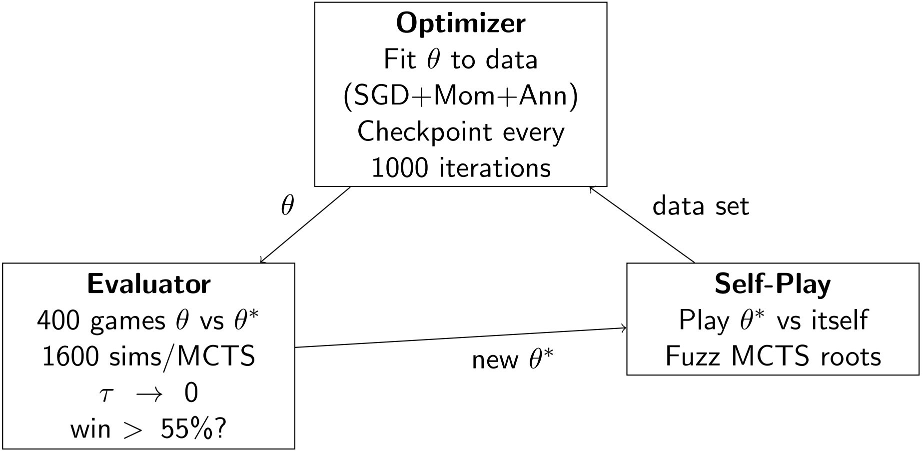
AlphaGo Zero MCTS



AlphaGo Zero MCTS



Training Pipeline



Takeaways

- Strongest player (5185 Elo)
- No prior knowledge beyond rules of game.
- And choice of architecture/hyper-parameters.
  (see blog of Oracle team reimplementation)

Outline

Setting

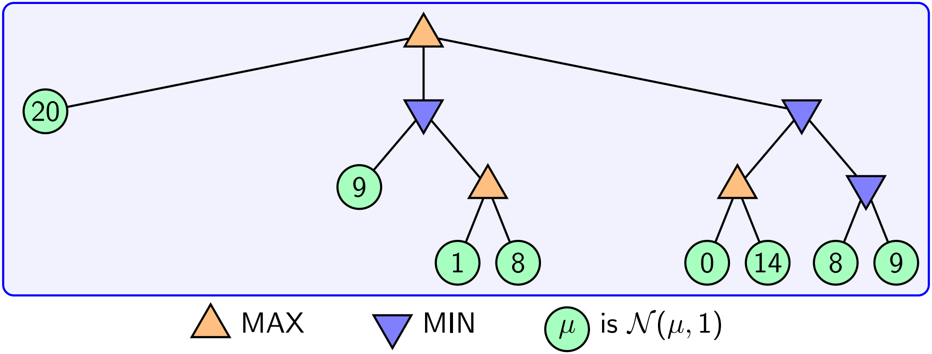
Main question

How should we allocate resources to learn fast (in games)?

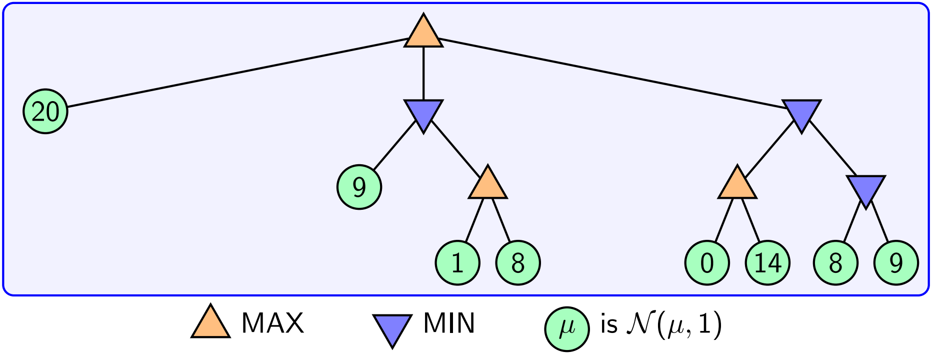
Gross stylisation of Game Tree Search problem:

- Limited resources to think strategically.
- Some process to estimate value at “tractability horizon”
- Here we assume noisy estimates (modelling rollout estimates)

Challenge Environment: Stochastic Game Tree Search



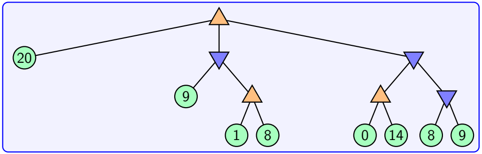
Challenge Environment: Stochastic Game Tree Search



Problem [Teraoka et al., 2014]

Determine the optimal move at the root
From **sample access** to the leaf payoffs

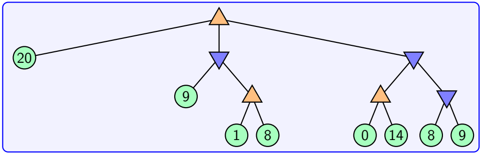
The Problem



Fix a game tree with payoff distributions in the leaves.

Optimal action at the root is a^*. Learning system sequentially takes samples and returns $\hat{a}$.

The Problem



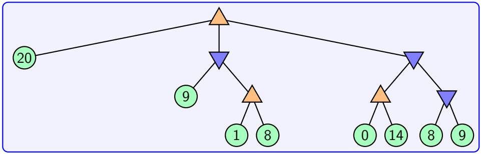
Fix a game tree with payoff distributions in the leaves.

Optimal action at the root is a^*. Learning system sequentially takes samples and returns $\hat{a}$.

Definition

Learner is δ-PAC when $\mathbb{P}(\hat{a} \neq a^*) \leq \delta$ for each instance.

The Problem



Fix a game tree with payoff distributions in the leaves.

Optimal action at the root is a^*. Learning system sequentially takes samples and returns $\hat{a}$.

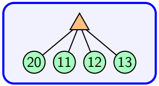
Definition

Learner is δ-PAC when $\mathbb{P}(\hat{a} \neq a^*) \leq \delta$ for each instance.

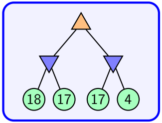
Goal: among δ-PAC learners, minimise sample complexity.

Note: active sequential multiple composite hypothesis test

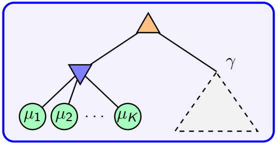
The Path Here



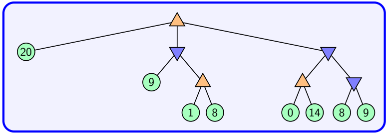
Best Arm Identification
[Garivier and Kaufmann, 2016]
Track-and-Stop



Depth 2
[Garivier et al., 2016]
Sparsity in lower bd



Depth 1.5
[Kaufmann et al., 2018]
Murphy Sampling



Any depth
[Degenne and Koolen, 2019]
Sticky Track-and-Stop

Lessons

- $\exists$ optimal allocation of effort among leaves [Castro, 2014, Garivier and Kaufmann, 2016].
- Often sparse.
- Convex program.
- Match with Track-and-Stop approach [Garivier and Kaufmann, 2016]
- Forced exploration ensures estimation.
- Discontinuity a real danger [Degenne and Koolen, 2019].
- Not hierarchical ???

Many questions remain open

- Practically efficient algorithms
- Remove forced exploration
- Moderate confidence $\delta \not\to 0$ regime [Simchowitz et al., 2017].
- Understand sparsity patterns
- Dynamically expanding horizon

- Active learning/testing beyond games

Thank you! And let's talk!