UC SANTA BARBARA

Using Off-Policy Self-Play Reinforcement Learning on Dots and Boxes: Designing Leagues For Performance

Erik Feng[1], João Hespanha[1]

1. UC Santa Barbara

1. Purpose

- Agent-to-agent self-play scales interaction and information to a level not possible with humans.
- The quality of the interactions of the agents remains crucial for sample-efficient learning.
- Using Dots and Boxes, a simple, scalable game, we design *leagues* [1] (i.e. groups or cohorts) to curate opponents that are more likely to provide high-quality interactions, and benchmark their performances.

2. Background and context

Dots and Boxes: A turn-based game where players draw lines between adjacent dots. Completing a box yields a square and an additional turn. The player with more squares when the game ends wins.

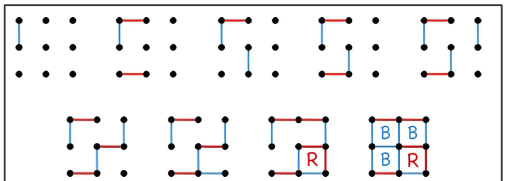
Research questions:

1. How does **league type** (PFSP variants vs pure self-play) affect learning?
2. Does model **capacity** (hidden size) matter at the step budgets we can afford?
3. How does **credit assignment** (**g=0** vs **g≠0**) change sample-efficiency and final strength?

Example League

ELO	Model	Sampling Probability
1100	model3	
1056	model1	
985	model2	
935	model0	

League samples stronger opponents more often



A typical 2-by-2 Dots and Boxes game

Image from Pencil And Paper Games

3. Methods and Math

- Using **5-by-5 Dots and Boxes games**
- **League design**[1] (PFSP variants vs self-play)
- **Credit assignment**: **g=0** vs **g≠0**
- Two **Deep-Q-Network MLPs** w/ hidden sizes ∈ {2, 16, [32,32]}.
- **League Settings**
  - *PFSP-var (even matches)* *PFSP-hard (hardest)*
  - *PFSP-target (70% percentile)* *Self sampling*
- **Opponent snapshot frozen for N episodes**
  - Reduces non-stationarity. (In most runs: freeze N=200, self-play 10–60%, random 5–20%.)
- **ε-greedy exploration:** ε decays to 0.05 on a fixed schedule.
- **Credit Assignment:**

Same player moves:

$$y = r + \gamma \max_{a' \in A_p(s')} Q_{\text{tgt},p}(s', a')$$

Opponent moves:

$$y = r - \gamma \max_{a' \in A_{1-p}(s')} Q_{\text{tgt},1-p}(s', a')$$

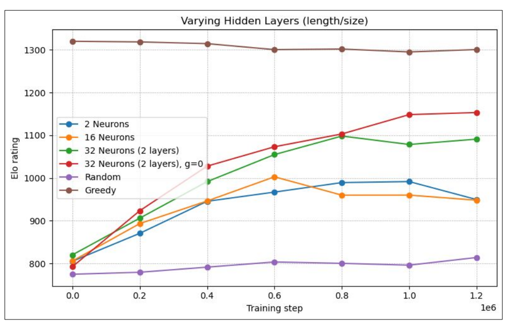
Baseline (g=0):

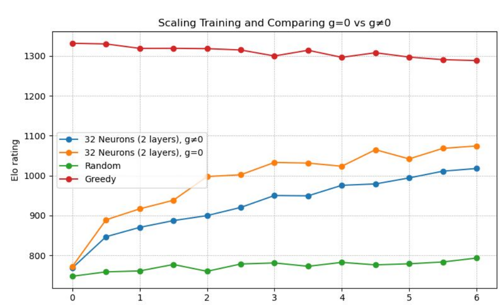
$$y = r$$

Double-DQN next-value[3] (for reward propagation and action selection):

$$a^* = \arg\max_{a'} Q_{\text{online,actor}}(s', a'), \quad q_{\text{next}} = Q_{\text{tgt,actor}}(s', a^*)$$

4. Graphs





5. Conclusions

League Type: League type seems not to make a difference in training performance. However, due to numerous optimizations, such as alpha-ramp, the leagues may not have been well-differentiated.

Model Capacity: Larger models tended to perform better, though this difference is easier to see with more training.

Credit assignment: In training, g=0 tended to increase performance. Whether if scaling training will reveal the short-sightedness of the model is still unknown.

Other Crucial Optimizations:

- Freezing Opponents for N episodes (after selecting from league) increases stability
- Increasing self-play early, then decreasing to increase league matches
- Alpha-ramp: start with g=0, then fade in sign-flip to learn greedy strategies quickly

6. Future Directions

Architecture changes: CNNs and GNNs provide more spatial information than MLPs.

League Management: Dynamic push and pops may allow for better training signals.

Train scaling: Scaling training to more steps may reveal weaknesses in myopic strategies (such as when g=0).

7. References

[1] Vinyals, O., et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," 2019 Nature.

[2] Lanctot, M., et al., "OpenSpiel: A Framework for Reinforcement Learning in Games," 2019 arXiv.

[3] Van Hasselt, H., Guez, A., & Silver, D., "Deep Reinforcement Learning with Double Q-Learning," 2016 AAAI.

AI Institute for Agent-based Cyber Threat Intelligence and Operation

Supported by the National Science Foundation under grant no. IIS-2229876 and is supported in part by funds provided by the NSF, by the Department of Homeland Security, and by IBM.

https://action.ucsb.edu