ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1.
Available from http://www.i6doc.com/en/.

Size Scaling in Self-Play Reinforcement Learning

Oren Neumann and Claudius Gros

Institute for Theoretical Physics
Goethe University Frankfurt
Frankfurt am Main, Germany

Abstract. Performance scaling laws with resources are heavily studied in supervised deep learning models but not in reinforcement learning. We examine the scaling of the AlphaZero [1] algorithm's performance with model size by training agents on three competitive two-player games, Connect Four, Oware and Pentago. We find that performance, in the form of the Elo rating, scales logarithmically with the number of free neural network parameters, a trend consistent across games and when using deeper neural networks. This leads to a universal expression for the average match outcome which depends only on the ratio of sizes between opponents, which is supported by an agnostic rating method.

1 Introduction

AlphaZero (αZ) [1] is the extension of AlphaGo Zero, which attained human expert level performance at playing the game of Go. Remarkably, αZ reached superhuman level for three different games (Go, Chess and Shogi) without any external knowledge by learning purely from self-play games. The algorithm, comprised of a tree search algorithm guided by a neural network (NN), received significant attention since its publication and several open source reconstructions of it have been released, most notably the one available by DeepMind's OpenSpiel [2]. While the original model required massive amounts of computation resources to train, training the algorithm on simpler games than the original three is possible on more modest hardware. However, performance scaling analysis for this framework has been lacking.

Scaling laws have received significant focus in recent years [3, 4, 5], in an effort to optimize the costs of training deep learning models, which can be in the millions of dollars for state-of-the-art models. Efforts focus mostly on language and computer vision tasks where training data is abundant and models are of proportional sizes. In contrast, reinforcement learning models tend to generate their own training data, making them less suitable for large scale optimization efforts.

In order to increase our understanding of performance scaling in reinforcement learning, we tracked the performance of αZ across different games and found a common scaling behavior for all cases considered, namely that the Elo rating scales logarithmically with the number of parameters in the NN, provided training is not bottlenecked by other factors. This behavior persists even when using an alternative rating scheme that does not contain the built-in biases of classical Elo rating.

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

2 Method

2.1 The AlphaZero algorithm

We trained agents with the αZ algorithm, which uses a combination of a Monte Carlo tree search (MCTS) guided by a deep NN. The agents are trained with reinforcement learning on self generated data, without using a priori knowledge of the game. The NN receives as input the current state s of the game and produces an output $f(s)$ that consists of a policy vector $\vec{p}$ and a value prediction v:

$$f(s) = (\vec{p}, v)\,, \tag{1}$$

where $v \in [-1, 1]$ is equivalent to the expected final outcome of the game when starting from the current position. The policy $\vec{p}$ is a probability distribution over available actions, with components $p_a = Pr(a|s)$ for each action a, that is used within αZ to guide the MCTS. The NN is trained on game states that were visited in previous self-play, using the loss function:

$$l = (z - v)^2 - \vec{\pi} \log \vec{p}\,. \tag{2}$$

Here z is the recorded game outcome and $\vec{\pi}$ is an improved policy vector generated by the MCTS. During a game, agents use the search tree for selecting their moves. For details see [1, 6].

2.2 Model training

We trained agents to play 2-player games using OpenSpiel [2], a collection of algorithms and environments for applying reinforcement learning to turn-based games. Training was done with OpenSpiel's implementation of the αZ algorithm together with various board game implementations. All agents used a multilayer perceptron (MLP) NN with two hidden layers (also three for Connect Four), coupled with a MCTS that probed a fixed number of 300 new states per turn.

Since our work focuses on general scaling phenomena, we avoided hyperparameter tuning as much as possible, by keeping the same hyperparamters used in the Python αZ example available on OpenSpiel. Only the following hyperparameters were changed, to allow for significant training within our compute budget: Replay buffer reuse was set to 10, NN model type to MLP and the checkpoint frequency to 10. The only parameters that varied between game types were the temperature drop and number of training steps: Temperature drop was set to 15 for Oware and to 5 for all other games, due to game length. The number of training steps was set to 2×10^3 for all cases except Connect Four with 3 layers, where we used 4×10^3 steps. Note that a training step occurs once 6.5×10^3 new game states are accumulated via self-play. All graphs were generated by averaging over 4 separately trained instances for each NN size.

2.3 Player rating

In order to measure agent performance we made use of the Elo rating system [7], a rating system for zero-sum games that was invented for chess and gained

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

popularity in numerous games. The expected score, the probability for player A to beat player B, is given by:

$$P_A = \frac{1}{1 + 10^{(R_B - R_A)/400}} , \tag{3}$$

where R_A, R_B are the Elo ratings of both players. These ratings are found by repeatedly matching players together and adjusting their ratings to fit the game outcomes.

Since Elo rating contains underlying assumptions about the form of the probability function, we used as well PolyRank [8], a method that simultaneously evaluates player rating and the probability distribution function. PolyRank applies polynomial regression to paired comparison data to estimate the comparison function via it's inverse, with convergence guaranteed for enough data and a large enough polynomial degree for the function estimation. We used polynomials of degree 9 for all cases.

3 Related work

Power law scaling of performance with available resources has been observed before in deep learning models. [3] found power law scaling for test loss with the number of model parameters, as well as with dataset size and training time, when training language models, and [5] found that the error of ResNets on ImageNet scales as a power law with FLOPs, in the low FLOPs regime.

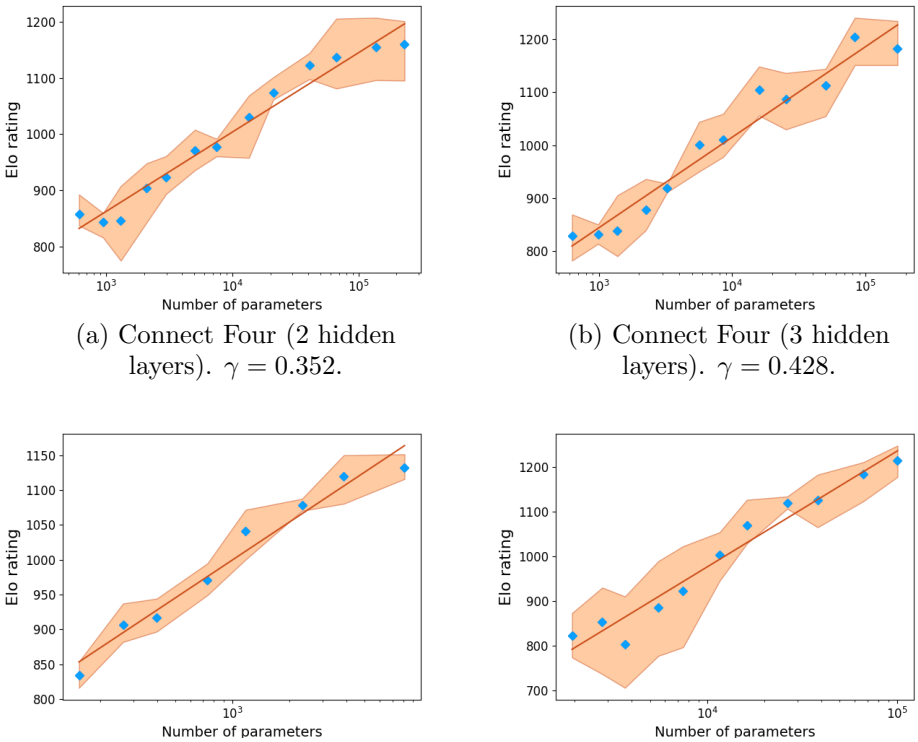
Performance scaling in reinforcement learning, and specifically for the αZ algorithm, has not received comparable attention. [6] investigated the performance of αZ for a 2-player knapsack problem and found logarithmic Elo scaling with model size. [9] used a variant of the algorithm for continuous action spaces and looked at the scaling of performance with the number of MCTS steps per move. [10] looked into the effects of different hyperparameters on training loss and Elo rating of αZ when learning to play the game Othello. To our knowledge no extensive study has been performed on the scaling of performance with model size of αZ or its predecessors AlphaGo and AlphaGo Zero.

4 Results

4.1 Elo rating

Our main results are presented in figure 1. As benchmarks, three games where chosen: Connect Four, Oware and Pentago. We trained AlphaZero models with fixed hyperparameters on all games, keeping a fixed MLP neural network architecture with two hidden layers (three layers as well for Connect Four) and varying only the width of the hidden layers. Four agents of identical architecture were used for each NN size, each trained only by self play and without access to external information. After training for a fixed number of training steps, each agent was matched against all other agents, which produced Elo ratings for the entire agent pool. Figure 1 shows the average Elo ratings of each NN size group,

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.



(a) Connect Four (2 hidden layers). $\gamma = 0.352$.

(b) Connect Four (3 hidden layers). $\gamma = 0.428$.

(c) Oware. $\gamma = 0.452$.

(d) Pentago. $\gamma = 0.648$.

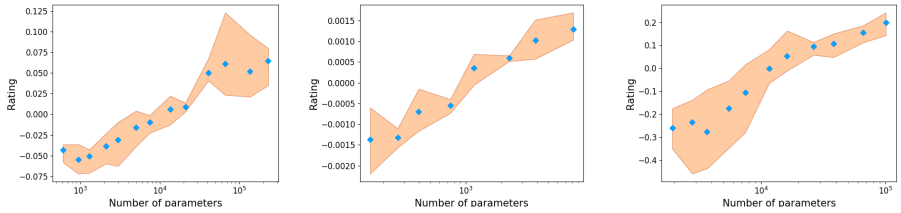
Fig. 1: Elo rating scales logarithmically with model size across different games (linear fit in logscale). Shaded area shows range of ratings among trained instances. The performance scaling exponent γ is cited for each case.

together with the range of ratings. A logarithmic trend is clearly visible in the Elo rating of the agents with respect to the number of free NN parameters. This implies a specific relation between the number of NN parameters used and the average outcome of the respective games, namely

$$P_A = \frac{n_A^\gamma}{n_A^\gamma + n_B^\gamma} = \frac{1}{1 + (n_B/n_A)^\gamma}\,, \tag{4}$$

where n_A, n_B are the numbers of parameters of agents A and B and P_A is the probability of agent A to win a match between them. The factor γ changes between games. We chose a fixed number of training steps for all agents, ensuring that it is long enough for the largest models to converge and that training is not bottlenecked by time. All other training hyperparameters were fixed, in order to ensure that the observed logarithmic trend represents solely the effects of a model size bottleneck.

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.



(a) Connect Four (b) Oware (c) Pentago

Fig. 2: PolyRank [8] rating scaling with model size. Shaded area show the range of ratings among trained instances.

Relation (4), which seems to be universal for zero-sum open-information games, implies that the statistics of game outcomes is determined under a model size bottleneck only by the ratio of NN sizes n_A/n_B, together with the factor γ, which is specific to the type of game played.

Two interesting behaviors result from this fact in different regimes: When $n_A \ll n_B$, the winning probability for agent A increases as a power of the relative resources used, here proportional to the ratio n_A/n_B of the numbers of parameters. In the case that both agents have NN sizes of a similar order of magnitude, scaling up the size relative to one's opponent, as $n_A = (1+\alpha)n_B$, will change the winning probability linearly when $\alpha \ll 1$, as $P_A = 1/(1+(1+\alpha)^{-\gamma}) \approx (1 + 0.5\alpha\gamma)/2$. This linear increase in winning probability is independent of n_A and n_B, providing a constant reward to a linear increase in model size regardless of the scale of network sizes.

It is worth noting that the logarithmic trend observed in figure 1 will eventually break for large models if the hyperparameters controlling learning are not adjusted with increasing model sizes, or when optimal play is reached for games like Connect Four, for which optimal strategies are known to exist. For this reason we fixed the hyperparameters, and specifically the number of training steps, such that training of the largest agents considered in our study converged. Training is therefore not bottlenecked by hyperparameter other than the number of NN parameters.

4.2 PolyRank rating

While Elo rating is the universal go-to rating system for assessing players of 2-player games, it contains assumptions that might not fit the actual data. Formula (3) can be reduced to the Bradley-Terry model [11]:

$$P_A = \frac{\mu_A}{\mu_A + \mu_B}\,, \tag{5}$$

where μ_i is a positive real-valued score assigned to player i. Using Elo rating one enforces the assignment of such a score to each player, for which we find the functional dependency $\mu_i = n_i^\gamma$. Although figure 1 shows a surprisingly good

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

fit of this probability model to our data, it is possible that the true underlying probability model is different. For this reason we used PolyRank [8] to rate our trained agents, a rating method that fits a probability model to the data while simultaneously rating the pool of agents. Figure 2 contains the resulting rating for all three games (all with NN models with 2 hidden layers). All three games still show a linear trend in logscale, albeit with a slight flattening for larger models, especially in the case of Connect Four.

5 Conclusion

We observed a clear scaling law for the AlphaZero algorithm which appears across different games and neural network (NN) depths. We showed that the logarithmic scaling of Elo ratings with NN size produces a simple expression for game outcome probabilities which depend solely on the ratio of NN sizes between opponents. An equivalent functionality was observed when using a rating system free from the bias of Elo rating. Our results suggest the existence of universal scaling laws in reinforcement learning and adversarial games, an area of research that currently lacks attention compared to the massive efforts to quantify performance scaling in NLP and computer vision. We hope these findings encourage others to use available open source resources to analyze hyperparameter tuning in reinforcement learning.

References

[1] David Silver et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[2] Marc Lanctot et al. OpenSpiel: A framework for reinforcement learning in games. *CoRR*, abs/1908.09453, 2019.

[3] Jared Kaplan et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[4] Jordan Hoffmann et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[5] Irwan Bello, William Fedus, Xianzhi Du, Ekin Dogus Cubuk, Aravind Srinivas, Tsung-Yi Lin, Jonathon Shlens, and Barret Zoph. Revisiting resnets: Improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 34:22614–22627, 2021.

[6] Oren Neumann and Claudius Gros. Investment vs. reward in a competitive knapsack problem. *arXiv preprint arXiv:2101.10964*, 2021.

[7] Mark E Glickman and Albyn C Jones. Rating the chess rating system. *CHANCE-BERLIN THEN NEW YORK-*, 12:21–28, 1999.

[8] Ivo FD Oliveira, Nir Ailon, and Ori Davidov. A new and flexible approach to the analysis of paired comparison data. *The Journal of Machine Learning Research*, 19(1):2458–2486, 2018.

[9] Thomas M Moerland, Joost Broekens, Aske Plaat, and Catholijn M Jonker. A0c: Alpha zero in continuous action space. *arXiv preprint arXiv:1805.09613*, 2018.

[10] Hui Wang, Michael Emmerich, Mike Preuss, and Aske Plaat. Hyper-parameter sweep on alphazero general. *arXiv preprint arXiv:1903.08129*, 2019.

[11] Rémi Coulom. Computing "elo ratings" of move patterns in the game of go. *ICGA journal*, 30(4):198–208, 2007.