SCALING LAWS FOR A MULTI-AGENT REINFORCEMENT LEARNING MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

The recent observation of neural power-law scaling relations has made a significant impact in the field of deep learning. A substantial amount of attention has been dedicated as a consequence to the description of scaling laws, although mostly for supervised learning and only to a reduced extent for reinforcement learning frameworks. In this paper we present an extensive study of performance scaling for a cornerstone reinforcement learning algorithm, AlphaZero. On the basis of a relationship between Elo rating, playing strength and power-law scaling, we train AlphaZero agents on the games Connect Four and Pentago and analyze their performance. We find that player strength scales as a power law in neural network parameter count when not bottlenecked by available compute, and as a power of compute when training optimally sized agents. We observe nearly identical scaling exponents for both games. Combining the two observed scaling laws we obtain a power law relating optimal size to compute similar to the ones observed for language models. We find that the predicted scaling of optimal neural network size fits our data for both games. We also show that large AlphaZero models are more sample efficient, performing better than smaller models with the same amount of training data.

1 INTRODUCTION

In recent years, power-law scaling of performance indicators has been observed in a range of machine-learning architectures (Hestness et al., 2017; Kaplan et al., 2020; Henighan et al., 2020; Gordon et al., 2021; Hernandez et al., 2021; Zhai et al., 2022), such as Transformes, LSTMs, Routing Networks (Clark et al., 2022) and ResNets (Bello et al., 2021). The range of fields investigated include natural language processing and computer vision (Rosenfeld et al., 2019). Most of these scaling laws regard the dependency of test loss on either dataset size, number of neural network parameters, or training compute. The robustness of the observed scaling laws across many orders of magnitude led to the creation of large models, with parameters numbering in the tens and hundreds of billions (Brown et al., 2020; Hoffmann et al., 2022; Alayrac et al., 2022).

Until now, evidence for power-law scaling has come in most part from supervised learning methods. Considerably less effort has been dedicated to the scaling of reinforcement learning algorithms, such as performance scaling with model size (Reed et al., 2022; Lee et al., 2022). At times, scaling laws remained unnoticed, given that they show up not as power laws, but as log-linear relations when Elo scores are taken as the performance measure in multi-agent reinforcement learning (MARL) (Jones, 2021; Liu et al., 2021) (see Section 3.2). Of particular interest in this context is the AlphaZero family of models, AlphaGo Zero (Silver et al., 2017b), AlphaZero (Silver et al., 2017a), and MuZero (Schrittwieser et al., 2020), which achieved state-of-the-art performance on several board games without access to human gameplay datasets by applying a tree search guided by a neural network.

Here we present an extensive study of power-law scaling in the context of two-player open-information games. Our study constitutes, to our knowledge, the first investigation of power-law scaling phenomena for a MARL algorithm. Measuring the performance of the AlphaZero algorithm using Elo rating, we follow a similar path as Kaplan et al. (2020) by providing evidence of power-law scaling of playing strength with model size and compute, as well as a power law of optimal model size with respect to available compute. Focusing on AlphaZero-agents that are guided by neural nets with fully connected layers, we test our hypothesis on two popular board games: Connect Four and

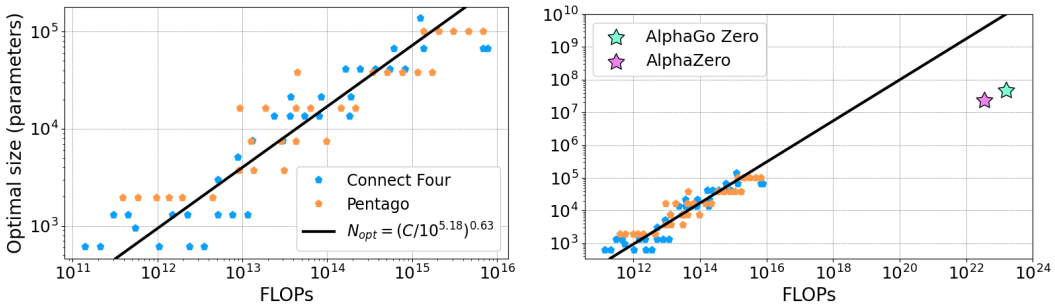

Figure 1: **Left:** Optimal number of neural network parameters for different amounts of available compute. The optimal agent size scales for both Connect Four and Pentago as a single power law with available compute. The predicted slope $\alpha_C^{opt} = \alpha_C/\alpha_N$ of Eq. (7) matches the observed data, where α_C and α_N are the compute and model size scaling exponents, respectively. See Table 3 for the numerical values. **Right:** The same graph zoomed out to include the resources used to create AlphaZero (Silver et al., 2017a) and AlphaGoZero (Silver et al., 2017b). These models stand well bellow the optimal trend for Connect Four and Pentago.

Pentago. These games are selected for being different from each other with respect to branching factors and game lengths.

Using the Bradley-Terry model definition of playing strength (Bradley & Terry, 1952), we start by showing that playing strength scales as a power law with neural network size when models are trained until convergence in the limit of abundant compute. We find that agents trained on Connect Four and Pentago scale with similar exponents.

In a second step we investigate the trade-off between model size and compute. Similar to scaling observed in the game Hex (Jones, 2021), we observe power-law scaling when compute is limited, again with similar exponents for Connect Four and Pentago. Finally we utilize these two scaling laws to find a scaling law for the optimal model size given the amount of compute available, as shown in Fig. 1. We find that the optimal neural network size scales as a power law with compute, with an exponent that can be derived from the individual size-scaling and compute-scaling exponents. All code and data used in our experiments are available online [1].

2 Related Work

Little work on power-law scaling has been published for MARL algorithms. Schrittwieser et al. (2021) report reward scaling as a power law with data frames when training a data efficient variant of MuZero. Jones (2021), the closest work to our own, shows evidence of power-law scaling of performance with compute, by measuring the performance of AlphaZero agents on small-board variants of the game of Hex. For board-sizes 3-9, log-scaling of Elo rating with compute is found when plotting the maximal scores reached among training runs. Without making an explicit connection to power-law scaling, the results reported by Jones (2021) can be characterized by a compute exponent of $\alpha_C \approx 1.3$, which can be shown when using Eq. 3. In the paper, the author suggests a phenomenological explanation for the observation that an agent with twice the compute of its opponent seems to win with a probability of roughly $2/3$, which in fact corresponds to a compute exponent of $\alpha_C = 1$. Similarly, Liu et al. (2021) report Elo scores that appear to scale as a log of environment frames for humanoid agents playing football, which would correspond to a power-law exponent of roughly 0.5 for playing strength scaling with data. Lee et al. (2022) apply the Transformer architecture to Atari games and plot performance scaling with the number of model parameters. Due to the substantially increased cost of calculating model-size scaling compared to compute or dataset-size scaling, they obtain only a limited number of data points, each generated by a single training seed. On this basis, analyzing the scaling behavior seems difficult. First indications of the model-size scaling law presented in this work can be found in Anonymized (2022).

[1] `https://github.com/AlphaZeroScaling/AlphaZero-Scaling`

3 BACKGROUND

3.1 LANGUAGE MODEL SCALING LAWS

Kaplan et al. (2020) showed that the cross-entropy loss of autoregressive Transformers (Vaswani et al., 2017) scales as a power law with model size, dataset size and compute. These scaling laws hold when training is not bottlenecked by the other two resources. Specifically, the model size power law applies when models are trained to convergence, while the compute scaling law is valid when training optimal-sized models. By combining these laws a power-law scaling of optimal model size with compute is obtained, with an exponent derived from the loss scaling exponents. These exponents, later recalculated by Hoffmann et al. (2022), tend to fall in the range $[0, 1]$.

3.2 POWER LAW SCALING AND ELO RATING

The Elo rating system (Elo, 1978) is a popular standard for benchmarking of MARL algorithms, as well as for rating human players in zero sum games. The ratings r_i are calculated to fit the expected score of player i in games against player j:

$$E_i = \frac{1}{1 + 10^{(r_j - r_i)/400}} \,, \tag{1}$$

where possible game scores are $\{0, 0.5, 1\}$ for a loss/draw/win respectively. This rating system is built on the assumption that game statistics adhere to the Bradley-Terry model (Bradley & Terry, 1952), for which each player i is assigned a number γ_i representing their strength. The player-specific strength determines the expected game outcomes according to:

$$E_i = \frac{\gamma_i}{\gamma_i + \gamma_j} \,. \tag{2}$$

Elo rating is a log-scale representation of player strengths (Coulom, 2007): $r_i = 400 \log_{10}(\gamma_i)$. An observed logarithmic scaling of the Elo score is hence equivalent to a power-law scaling of the individual strengths: If there exists a constant c such that $r_i = c \cdot \log_{10}(X_i)$ for some variable X_i (e.g. number of parameters), then the playing strength of player i scales as a power of X_i:

$$\gamma_i \propto X_i^{\alpha} \,, \tag{3}$$

where $\alpha = c/400$. Note that multiplying γ_i by a constant is equivalent to adding a constant to the Elo score r_i, which would not change the predicted game outcomes. This power law produces a simple expression for the expected result of a game between i and j:

$$E_i = \frac{1}{1 + (X_j/X_i)^{\alpha}} \,. \tag{4}$$

4 EXPERIMENTAL SETTING

We train agents with the AlphaZero algorithm using Monte Carlo tree search (MCTS) (Coulom, 2006; Browne et al., 2012) guided by a multilayer perceptron. Kaplan et al. (2020) have observed that autoregressive Transformer models display the same size scaling trend regardless of shape details, provided the number of layers is at least two. We therefore use a neural network architecture

Table 1: Summary of scaling exponents. α_N describes the scaling of performance with model size, α_C of performance with compute, and α_C^{opt} of the optimal model size with compute. Connect Four and Pentago have nearly identical values.

Exponents	Connect Four	Pentago
α_N	0.88	0.87
α_C	0.55	0.55
α_C^{opt}	0.62	0.63

where the policy and value heads, each containing a fully-connected hidden layer, are mounted on a torso of two fully-connected hidden layers. All hidden layers are of equal width, which is varied between 4 and 256 neurons. Training is done using the AlphaZero Python implementation available in OpenSpiel (Lanctot et al., 2019).

In order to make our analysis as general as possible we specifically avoid hyperparameter tuning whenever possible, fixing most hyperparameters to the ones suggested in OpenSpiel's AlphaZero example code, see Appendix A. The only parameter tailored to each board game is the temperature drop, as we find that it has a substantial influence on agents' ability to learn effectively within a span of 10^4 training steps, the number used in our simulations.

We focus on two different games, Connect Four and Pentago, which are both popular two-player zero-sum open-information games. In Connect Four, players drop tokens in turn into a vertical board, trying to win by connecting four of their tokens in a line. In Pentago, players place tokens on a board in an attempt to connect a line of five, with each turn ending with a rotation of one of the four quadrants of the board. These two games are non-trivial to learn and light enough to allow for training a larger number of agents with a reasonable amount of resources. Furthermore, they allow for the benchmarking of the trained agents against solvers that are either perfect (for Connect Four) or near-to-perfect (for Pentago). Our analysis starts with Connect Four, for which we train agents with varying model sizes. Elo scores are evaluated both within the group of trained agents and with respect to an open source game solver (Pons, 2015). We follow up the Connect Four results with agents trained on Pentago, which has a much larger branching factor and shorter games (on average, see Table 2).

For both games we train AlphaZero agents with different neural network sizes and/or distinct compute budgets, repeating each training six times with different random seeds to ensure reproducibility (Henderson et al., 2018). We run matches with all trained agents, in order to calculate the Elo ratings of the total pools of 1326 and 714 agents for Connect Four and Pentago, respectively. Elo score calculation is done using BayesElo (Coulom, 2008).

5 Results

5.1 Neural network size scaling

Fig. 2 (top) shows the improvement of Connect Four Elo scores when increasing neural network sizes across several orders of magnitude. We look at the infinite compute limit, in the sense that performance is not bottlenecked by compute-time limitations, by training all agents exactly 10^4 optimization steps. One observes that Elo scores follow a clear logarithmic trend which breaks only when approaching the hard boundary of perfect play. In order to verify that the observed plateau in Elo is indeed the maximal rating achievable, we plot the Elo difference between each agent and an optimal player guided by a game solver (Pons, 2015) using alpha-beta pruning (Knuth & Moore, 1975). A vertical line in both plots marks the point beyond which we exclude data from the fit. This is also the point where agents get within 10 Elo points difference from the solver.

Combining the logarithmic fit with Eq. 1 yields a simple relation between the expected game score E_i for player i against player j and the ratio N_i/N_j, of the respective numbers of neural network parameters:

$$E_i = \frac{1}{1 + (N_j/N_i)^{\alpha_N}} \,, \tag{5}$$

Table 2: Game details. Connect Four games last longer on average, but the branching factor of Pentago is substantially larger. Game lengths are averaged over all training runs, citing the maximal allowed number of turns as well.

Game	Branching Factor	Average Game Length	Maximal Game Length
Connect Four	≤ 7	25.4	42
Pentago	≤ 288	16.8	36

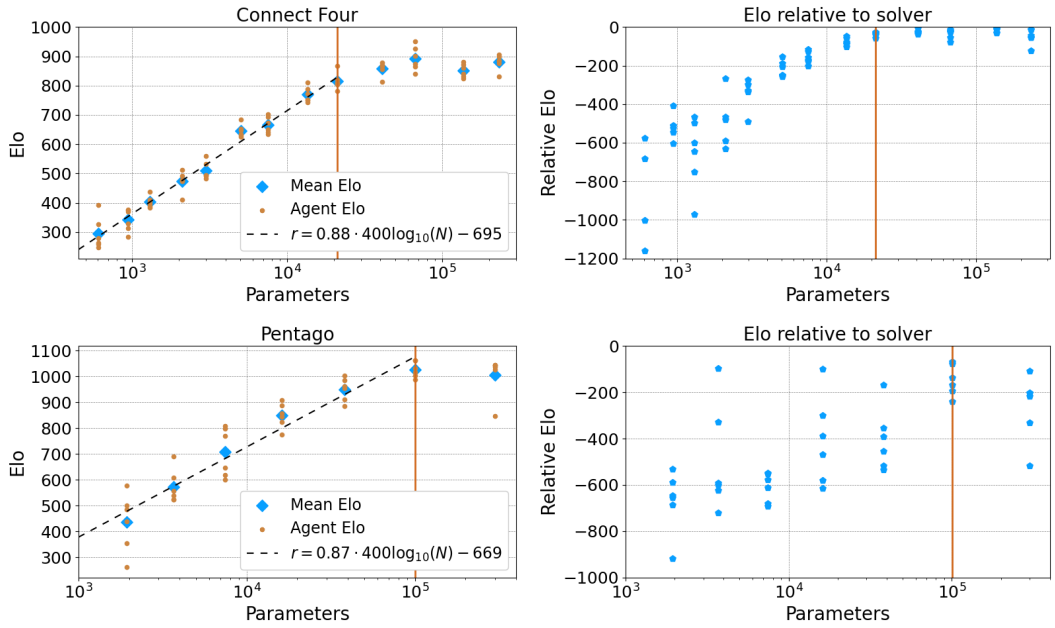

Figure 2: AlphaZero performance scaling with neural-network parameters. **Top left:** Agents trained on Connect Four. Elo scores follow a clear linear trend in log scale that breaks off only when approaching the optimal playing strategy. This can can be seen using a game solver (top right). **Top right:** Elo scores obtained by only considering matches against an optimal player, created with a game solver (Pons, 2015). Agents to the right of the vertical line are within ten Elo-points distance from the perfect solver. Increased data scattering is due to Elo scores being calculated against a single benchmark player. **Bottom left:** The Elo scores of Pentago as a function of model size follow the same trend as Connect Four. Improvement flattens for large parameter counts when performance approaches the level of a benchmark agent aided by an optimal play database (Irving, 2014) (bottom right).

where the power α_N is a constant determined by the slope of the fit. In the context of the Bradley-Terry model, Eq. (2), this means that player strength follows a power law in model size, $\gamma \propto N^{\alpha_N}$.

Results for the game of Pentago are presented at the bottom of Fig. 2. Pentago similarly exhibits log scaling of average agent Elo scores as a function of model size that flattens out only with a high number of neural network parameters. The scaling exponent is strikingly close to that of Connect Four (see Table 1), suggesting this exponent might be characteristic for a certain, yet unknown class of games. Again, in order to verify that the observed plateau is due to approaching the boundary of optimal play, we plot the Elo difference between each agent and a close to optimal benchmark player aided by a game solver. The bottom right of Fig. 2 shows that large agents are much closer to the level of the benchmark player. The benchmark player is based on the best performing Pentago agent trained, given 10 times more MCTS steps per move. We further improve it by constraining all moves before ply 18 to be optimal, using a 4TB dataset created by a Pentago solver program (Irving, 2014). A vertical line in both plots marks the point the log trend breaks, where agents approach the level of the benchmark player. Similar to Connect four we exclude the largest agent size beyond that point from the fit where adding more parameters no longer increases performance.

The power-law scaling of the playing strength of MARL agents can be considered analogous to the scaling laws observed for supervised learning. A noteworthy difference is the type of performance parameter being scaled. Scaling laws for language models tend to scale cross-entropy test loss with resources, which has been correlated to improved performance on downstream tasks (Hoffmann et al., 2022). In comparison, Eq. 5 shows power law scaling of player strength rather than test loss, which is directly related to the Elo score, a common measure of task performance in MARL. In fact, neural network loss on a test set generated by a game solver are observed to be a poor predictor of performance scaling both for the value head and the policy head, see Appendix F.

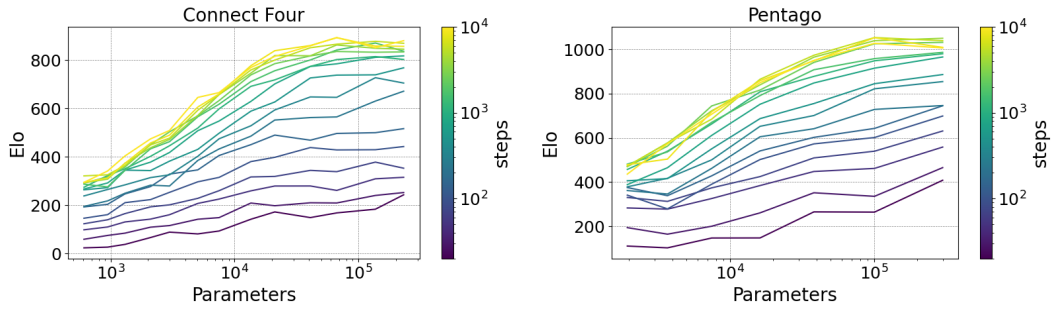

Figure 3: Elo scaling with parameters during training. For both games, log-linear scaling appears already at early stages of training, albeit with lower slopes. Elo curves converge for large numbers of training steps when agents approach their maximal potential.

5.2 Effect of training time on size scaling

Similar to language model scaling laws, the scaling exponent of Eq. 5 only applies at the limit of infinite compute when training is not bottlenecked by the number of training steps. Fig. 3 shows how performance scaling is influenced by the amount of available compute in terms of training steps. One observes that the Elo score scales logarithmically with model size for all training regimes when performance is not affected, as before, by the perfect play barrier. The slope increases with the number of training steps, but with diminishing returns, see Appendix D. For both games one observes that Elo scores converge to a limiting curve when the compute budget becomes large. There is hence a limiting value for the scaling exponent α_N, as defined by Eq. (5).

5.3 Compute scaling

To examine compute scaling we plot Elo scores against training self-play compute, defined here by $C = S \cdot T \cdot F \cdot D$, where S is the number of optimization steps, T is the number of MCTS simulations per move, F the compute cost of a single neural network forward pass, and D is the amount of new datapoints needed to trigger an optimization step. We do not count the compute spent during optimization steps, which is much smaller than the compute needed to generate training data. In our case we have approximately 2000 forward passes for each backward pass. We also approximate the number of MCTS simulations T to be the maximum number of simulations allowed, since the maximum is reached at all game positions except late-game positions, where the remaining game tree is already fully mapped.

Fig. 4 shows the Elo scores of agents with different training compute measured in units of floating-point operations (FLOPs). The Pareto front, which is the group of all agents with maximum Elo among agents with similar or less compute, follows a log-linear trend up to the point of optimal play. Together with Eq. 1 this yields an expected game score of:

$$E_i = \frac{1}{1 + (C_j/C_i)^{\alpha_C}} \,, \tag{6}$$

for agents i, j with training computes C_i, C_j and optimal neural network sizes. Player strength scales with the same exponent, $\gamma \propto C^{\alpha_C}$. This scaling behaviour also holds for the average agent performance when we plot the Pareto front of compute curves averaged over multiple training runs with different random seeds (see Appendix D). Eq. 6 therefore applies not only to the best case training scenario, but also when training a single agent instance.

As with the size scaling exponent, Connect Four and Pentago have similar compute scaling exponents (see Table 1). We note that our training compute exponents are significantly smaller than the compute exponent obtained when applying Eq. 6 to the data obtained by Jones (2021) for small-board Hex, which would be roughly $\alpha_C = 1.3$. The reason for this difference is at present unclear. Possible explanations could be the different game types, or the choice of neural net architecture; while we keep the depth constant, Jones (2021) increase both depth and width simultaneously.

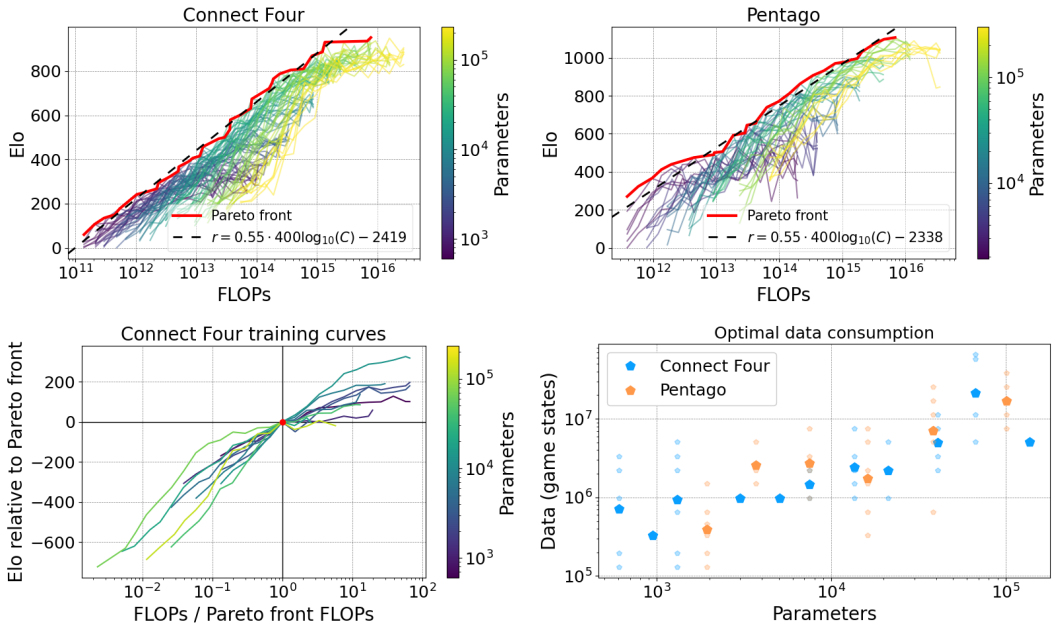

Figure 4: **Top:** Performance scaling with training compute. The Pareto front follows a log-linear trend in compute right up to the optimal play plateau, both for Connect Four and Pentago (left/right panel). **Bottom left:** Average Connect Four training curves relative to the point at which they reach the Pareto front. Agents continue to improve significantly after passing through the Pareto optimal point, meaning optimal training should stop long before convergence (compare to the training curves in Silver et al. (2017b;a)). **Bottom right:** The amount of data required to reach the compute Pareto front increases as the parameter count increases. Large dots are the geometric mean of data usage on the Pareto front.

5.4 Estimating the compute optimal model size

We now turn to our central result, efficiency scaling. Scaling laws are attracting considerable attention as they provide simple rules for the optimal balancing between network complexity and computing resources. The question is, to be precise, whether we can predict how many neural network parameters one should use when constrained by a certain compute budget in order to get an AlphaZero agent with a maximal Elo score.

The agents on the Pareto front shown in Fig. 4 have the highest Elo score for all agents with the same or lower compute budget. In order to find the relationship between optimal network size and the given compute budget, we examine the network size of the Pareto agents and the amount of FLOPs they consumed in training. This is shown in Fig. 1, which contains an aggregate of all compute-optimal agents trained on Connect Four and Pentago. We see a clear power law trend common to both games of the form:

$$N_{opt}(C) = \left(\frac{C}{C_0}\right)^{\alpha_C^{opt}}, \tag{7}$$

where N_{opt} is the optimal neural network size and α_C^{opt}, C_0 are constants.

Next we point out that the optimality scaling exponent α_C^{opt} can be calculated analytically given the scaling laws we demonstrated for parameter count and compute. For this we use Eq. (5) and Eq. (6). Based on the fact that the Bradley Terry playing strength γ scales as $\lim_{C\to\infty}\gamma(N, C) \propto N^{\alpha_N}$ at the limit of infinite compute, and as $\gamma(N_{opt}, C) \propto C^{\alpha_C}$ along the curve of optimal sizes, we can combine the two to obtain the constraint:

$$\alpha_C^{opt} = \frac{\alpha_C}{\alpha_N}. \tag{8}$$

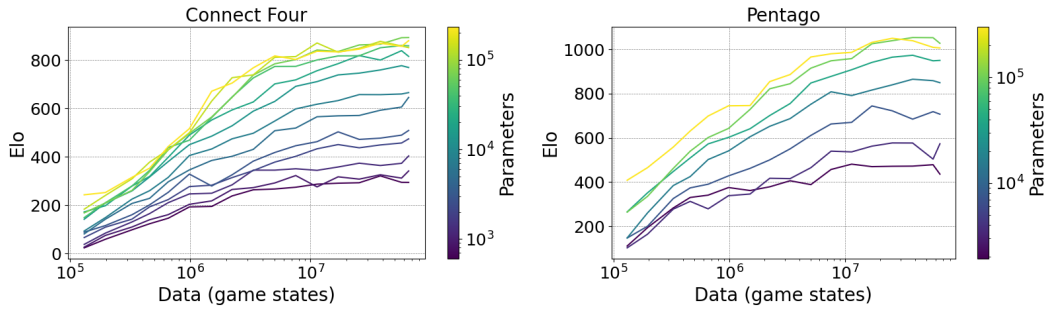

Figure 5: Performance improvement with data (game states) generated. Model sizes are color coded. Larger models are more sample efficient, achieving better Elo scores than smaller models when training on the same amount of game states.

This relationship is obtained when $N = N_{opt}$ and $C \to \infty$, but holds for every choice of N or C since the exponents are constants. The scale constant C_0 is calculated empirically. We use the scaling exponents α_N and α_C obtained by fitting the Elo ratings for Connect Four and Pentago to get α_C^{opt} via Eq. (8). The predicted scaling law is included in Fig. 1 fitting only C_0. The predicted exponent fits the data remarkably well.

5.4.1 Extrapolating to state of the art models

For comparison, Fig. 1 right shows where AlphaGo Zero, trained by Silver et al. (2017b), and AlphaZero, trained by Silver et al. (2017a), would stand relative to the power law specified by Eq. (8). We base our estimates for the number of parameters from the respective papers, and use the training compute values estimated by Sevilla et al. (2022) and Amodei et al. (2018) for these models. We find that these models fall well below the optimal curve predicted from scaling MLP agents playing Connect Four and Pentago.

The scaling law plotted in Fig. 1 might not accurately describe AlphaGo Zero and AlphaZero trained on Go, chess and shogi due to the differences between the games: these games are much larger in observation size and branching factor than Connect Four and Pentago, as well as having different win conditions and game mechanics. The use of ResNets rather than MLPs can also be significant. It remains to future study to determine if these models exhibit related scaling trends. In the case that Eq. (8) does apply to agents using ResNets and trained on Go, chess, or shogi, then a better Elo score could have been achieved with the same training time by training larger neural nets by up to two orders of magnitude. A possible indication that these models were too small for the amount of compute spent training them can be seen from the training figures of Silver et al. (2017b;a); the AlphaGo Zero and AlphaZero training curves both have long tails with minimal improvement at late stages of training. Our analysis shows that optimal training for a fixed compute budget will stop long before performance has converged, in agreement with language model scaling laws (Kaplan et al., 2020), compare Fig. 4 bottom left. That being said, a reason to favour smaller models is their lower inference-time compute cost, which can be critical in the case of a time limit on move selection. We elaborate on inference compute scaling in Appendix B.

5.5 Sample efficiency

Unlike supervised learning models where dataset size is an important metric due to the cost of collecting it, AlphaZero is not constrained by data limitations since it generates its own data while training. It is however important to understand how data requirements change as it provides us an insight into other sub-fields of reinforcement learning where data generation is costly, for example in robotics models, which need real life interactions with the environment in order to gather training data (Dulac-Arnold et al., 2021).

For this reason we look at performance scaling with data for different sized models, as shown in Fig. 5. We observe that agents improve at different rates with the amount of training data, which is defined here by the number of game states seen during self-play. Specifically, agents with larger neu-

ral nets consistently outperform smaller agents trained on the same amount of data, a phenomenon of data efficiency that has been observed in language models (Kaplan et al., 2020). However, when training is not bottlenecked by the amount of training data, an opposite trend appears: the amount of data required to reach optimal performance under a fixed compute budget increases with model size, as is visible in the bottom right panel of Fig. 4.

6 DISCUSSION

In this work we provided evidence to the existence of power-law scaling for the performance of agents using the AlphaZero algorithm on two games, Connect Four and Pentago. For both games one observes scaling laws qualitatively paralleling the ones found in neural language models. We pointed out that a log-scaling of Elo scores implies a power law in playing strengths, a relation that is captured by Eq. (4) and that can be used to predict game results. We showed that AlphaZero agents that are trained to convergence, with no limit on available compute, achieve Elo scores that are proportional to the logarithm of the respective number of neural network parameters. We demonstrated this for two games with significantly different branching parameters and tyical game lengths, finding that they share similar scaling exponents for playing strength. This result indicates the possible existence of universality classes for performance scaling in two-player open information games.

We then demonstrated that Elo scores scale logarithmically with available compute when training optimally sized agents, again finding similar scaling exponents for the two games studied. We showed that the optimal number of neural network parameters scales as a power of compute with an exponent that can be derived from the other two scaling laws. This scaling model is especially important as it allows one to achieve maximal performance within the limits of a given training budget by using an optimally sized neural net. We showed optimal training stops long before convergence, in contrast to what was done with state-of-the-art models. Finally we observed that agents with larger neural nets are always more data efficient than smaller models, which could be important in particular for reinforcement learning studies with expensive data generation.

We find it note worthy that scaling laws that are common to language and other supervised learning models are also present in one of the most important MARL models. This scaling behavior could be common to other reinforcement learning algorithms, which would provide an opportunity to optimize their resource allocation. We note that the main significance of the scaling exponents found in this study is not their reported value, but the evidence of power-law scaling. Consider the scaling exponents found by Kaplan et al. (2020), which were later corrected by optimizing the learning rate schedule (Hoffmann et al., 2022). The tuning of AlphaZero's hyperparameters could affect the observed exponents in a similar manner. Particularly important hyperparameters are the number of states explored by the MCTS, the frequency of optimization steps, and the update frequency of the data generating model. We leave the investigation of hyperparameter impact to future work.

As a side remark, we would like to comment on the relevance of scaling laws for studies without massive computing budgets. A rising concern since the discovery of performance scaling laws regards the resulting push towards substantially increased model sizes. This decreases the impact of research utilizing smaller-sized models, the bread-and-butter approach of smaller research groups. (Strubell et al., 2019). On the other hand, the existence of scaling laws implies that it is possible to gain relevant insights into the behaviour of large models by training medium-sized models, the case of the present work. Interestingly, our initial test runs were performed on a toy problem that involved training agents on a desktop computer with just four CPU cores. Even with such a minimal compute budget a clear scaling trend emerged (Anonymized, 2021). We believe our work shows that significant insights can be obtained at times also with small-scale analysis.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Dario Amodei, Danny Hernandez, Girish SastryJack, ClarkGreg Brockman, and Ilya Sutskever. Ai and compute, 2018. URL `openai.com/blog/ai-and-compute/`.

Anonymized. Anonymized. *arXiv preprint*, 2021.

Anonymized. Anonymized. *Anonymized*, 2022.

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.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. In *International Conference on Machine Learning*, pp. 4057–4086. PMLR, 2022.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006.

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

Rémi Coulom. Whole-history rating: A bayesian rating system for players of time-varying strength. In *International Conference on Computers and Games*, pp. 113–124. Springer, 2008.

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

Arpad E Elo. *The rating of chessplayers, past and present*. Arco Pub., 1978.

Mitchell A Gordon, Kevin Duh, and Jared Kaplan. Data and parameter scaling laws for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5915–5922, 2021.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Patwary, Mostofa Ali, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Geoffrey Irving. Pentago is a first player win: Strongly solving a game using parallel in-core retrograde analysis. *arXiv preprint arXiv:1404.0743*, 2014.

Andy L Jones. Scaling scaling laws with board games. *arXiv preprint arXiv:2104.03113*, 2021.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Donald E Knuth and Ronald W Moore. An analysis of alpha-beta pruning. *Artificial intelligence*, 6 (4):293–326, 1975.

Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, Daniel Hennes, Dustin Morrill, Paul Muller, Timo Ewalds, Ryan Faulkner, János Kramár, Bart De Vylder, Brennan Saeta, James Bradbury, David Ding, Sebastian Borgeaud, Matthew Lai, Julian Schrittwieser, Thomas Anthony, Edward Hughes, Ivo Danihelka, and Jonah Ryan-Davis. OpenSpiel: A framework for reinforcement learning in games. *CoRR*, abs/1908.09453, 2019. URL `http://arxiv.org/abs/1908.09453`.

Kuang-Huei Lee, Ofir Nachum, Mengjiao Yang, Lisa Lee, Daniel Freeman, Winnie Xu, Sergio Guadarrama, Ian Fischer, Eric Jang, Henryk Michalewski, et al. Multi-game decision transformers. *arXiv preprint arXiv:2205.15241*, 2022.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, SM Eslami, Daniel Hennes, Wojciech M Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, et al. From motor control to team play in simulated humanoid football. *arXiv preprint arXiv:2105.12196*, 2021.

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.

Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean-Baptiste Lespiau, Wojciech M Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. α-rank: Multi-agent evaluation by evolution. *Scientific reports*, 9(1):1–29, 2019.

Pascal Pons. Connect four solver, 2015. URL `gamesolver.org`.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*, 2019.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34:27580–27591, 2021.

Jaime Sevilla, Lennart Heim, Anson Ho, Tamay Besiroglu, Marius Hobbhahn, and Pablo Villalobos. Compute trends across three eras of machine learning. *arXiv preprint arXiv:2202.05924*, 2022.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017b.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12104–12113, 2022.

A HYPERPARAMETERS

In order to keep our results as general as possible, we try to avoid hyperparameter tuning and choose to train all agents with the values suggested by OpenSpiel's Python AlphaZero example. Table 3 contains a summary of all hyperparameters used and their meaning.

Table 3: Training hyperparameters used for Connect Four, Pentago and Oware.

Hyperparameter	Value	Description
c_{uct}	2	MCTS exploration constant
Max simulations	300	Number of MCTS steps to run per move
Batch size	2^{10}	Batch size during optimization steps
Replay-buffer size	2^{16}	Number of game states stored in the replay buffer
Replay-buffer reuse	10	Number of optimization steps a game state stays in the buffer before it is erased
Learning rate	0.001	Optimization learning rate
Weight decay	0.0001	L2 regularization strength
Policy ϵ	0.25	Policy noise
Policy α	1	Dirichlet noise variable
Temperature	1	Temperature applied to the policy for final move selection
Temperature drop	varied	Number of turns before temperature is set to 0

Temperature drop is the only parameter with a different value in each game, set to 15 and 5 for Connect Four and Pentago, respectively. We do this since we find that this term, which should correlate with game length, can drastically affect the performance of agents when we train them for 10^4 optimization steps.

Replay-buffer reuse is set to 10 rather than 3, the value suggested in the original example, to speed up training by leaving states for longer in the replay buffer (Lillicrap et al., 2015).

During matches, all noise is set to zero but temperature is set to 0.25, with an infinite *temperature drop*, meaning temperature stays constant throughout the game. We do this since without a random element all games between two agents will be identical. A high temperature would distort the data by reducing the Elo difference between players, which is why we use a low temperature value of 0.25 that produces roughly 5% or less repeating games. The number of MCTS simulations per move is the same for all agents at test time, equal to the number used during training.

B INFERENCE-TIME COMPUTE SCALING

Reinforcement learning models like AlphaZero are often required to perform fast at test time, either in order to finish their calculation within a thinking time limit or to react quickly to a rapidly changing environment. The AlphaZero algorithm provides the ability to tailor the move-selection inference time to a given time limit by changing the number of MCTS steps, which does not have to

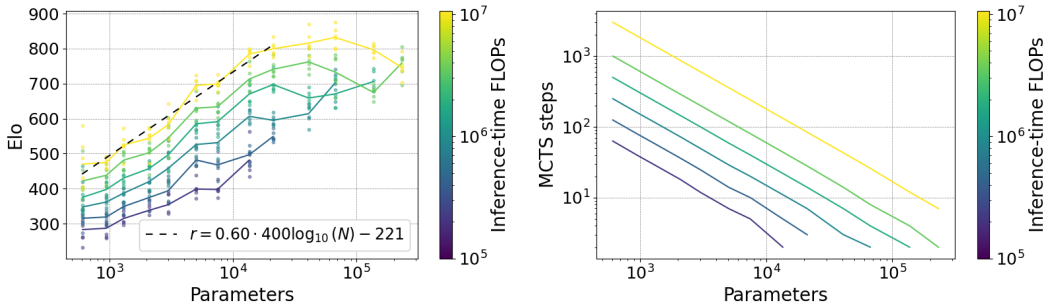

Figure 6: **Left:** Connect Four performance at fixed values of inference-time compute, enforced by scaling MCTS search size inversely with forward-pass compute costs. Elo scores steadily improve with increasing model size for smaller models, but the trend breaks for large models. **Right:** The number of MCTS steps used by different sized models with different inference-time compute budgets. Agents with less than 2 MCTS steps per turn are excluded from both plots.

match the number of steps used during training. Since our main results all use a fixed number of 300 MCTS steps both at train and test time, we add here the scaling behavior of different sized Connect Four agents using varying numbers of MCTS steps. We look at the case where training compute is abundant but performance is bottlenecked by inference-time compute. To do so we match fully converged agents against each other, all receiving a fixed amount of inference-time compute enforced by scaling MCTS search size inversely with forward-pass compute costs.

Fig. 6 shows the Elo scaling of different sized agents under inference-time compute constraints. Agents steadily improve with size at smaller sizes for all amounts of available inference-time compute. Larger agents break off from this trend at different points which depend on the amount of compute given, such that the the trend seems to hold longer for larger compute values. Interestingly, a power law appears at the limit of large inference-time compute which breaks off only for the largest agents. The scaling exponent is significantly smaller than that of Fig. 2, which is not surprising given that small agents have access to up to two orders of magnitude more MCTS steps than the largest agents, see Fig. 6 right. This scaling law may be related to the power law scaling of inference-time compute with training compute found by Jones (2021) when fixing the Elo score of AlphaZero agents.

C Measuring Elo against other players

Elo rating is a system that generates predictions of game outcomes that are based on ground truth values. As such, it is vulnerable to biases in the underlying dataset of game outcomes. One example of a potential bias is when all players use a restricted set of strategies, resulting in Elo scores that will not necessarily represent the interactions of these players with agents utilizing different sets of strategies. Here we provide evidence that the scaling laws presented in this paper are not the result of such biases.

Although Fig. 2 right presents Elo scores calculated solely from matches between agents and a perfect Connect Four game solver, these scores are only accurate close to the solver score and are unreliable for agents far from optimal play. For example, the worst agent is almost 1,200 Elo points away from the solver, meaning it should win about one game in a 1,000. This requires a large amount of matches in order to model accurately, and indeed several agents are missing from this figure since they received an Elo score of minus infinity, never winning a single match.

Here we solve this issue by producing a set of solver agents of varying playing strengths, matching them with each other and with AlphaZero agents trained on Connect Four to produce Fig. 7. All solver agents base their prior on the vector $\boldsymbol{q}$ provided by the game solver, which contains information about the quality of all legal moves. Taking action a in game state s, the number $\boldsymbol{q}(s, a)$ equals to 42 minus the minimal number of turns until either player can force a victory under perfect play, with a minus sign if the current player would lose. See Pons (2015) for a detailed explanation of how $\boldsymbol{q}$ is generated. The final prior $\boldsymbol{\pi}$ is created by applying a softmax to $\boldsymbol{q}$ divided by a temperature

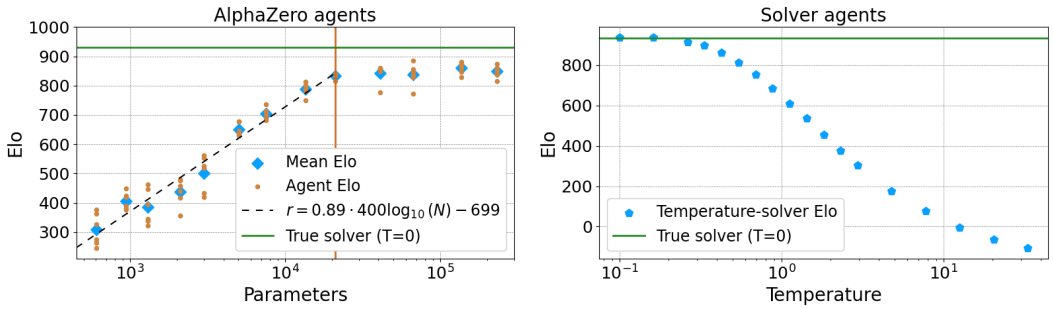

Figure 7: **Left:** AlphaZero size scaling with Elo scores calculated by only using matches between solver agents and AlphaZero agents, or matches between two solvers. The scaling behavior of Fig. 2 holds here as well, with an almost identical scaling exponent. Since the true solver, with temperature set to zero, is part of the agent pool, it has a well defined Elo score (green line). **Right:** Solver-agent scaling with temperature. Each solver agent uses the prior produced by the Connect Four game solver (Pons, 2015), applying a temperature transformation to it according to Eq. 9. Low temperature agents asymptotically approach the true solver Elo score (green line), while high temperature agents converge to the score of a random player. The range of Elo scores covers that of AlphaZero agents, allowing an accurate measurement of their Elo scores.

value T:

$$\boldsymbol{\pi}(s, a) = \frac{e^{\mathbf{q}(s,a)/T}}{\sum_b e^{\mathbf{q}(s,b)/T}} \,. \tag{9}$$

This provides a smooth interpolation between the optimal prior and a uniform prior. At $T = 0$ the prior leads to the fastest victory if possible, otherwise to a draw if possible, and if neither then to the slowest defeat. At the limit $T \to \infty$ the prior is uniform, generating a completely random agent. Note that T is *not* the temperature mentioned in Table 3, which is set to zero for the solver agents and to 0.25 for AlphaZero agents, as it is done throughout this paper, see Appendix A.

Fig. 7 shows the Elo scaling calculated with solver agents. Unlike in Fig. 2, Elo scores are calculated exclusively from data on matches between AlphaZero agents and solver agents as well as matches between pairs of solver agents. No game between two AlphaZero agents was included in this calculation. The resulting Elo scores scale as a log of model size with an almost identical exponent to that found in Table 1, providing evidence that this scaling behavior is universal and not the product of a bias in the choice of players.

We note that while this way of measuring Elo strengthens our results by reproducing the scaling exponents, it is probably a less accurate measure than the one used in Section 5.1. This is because all solver agents use the same strategy, which introduces bias into the match outcome dataset. This could be the reason why the spread of agent Elo scores in Fig. 7 is slightly wider than that of Fig. 2 for smaller agents. Despite this bias, the model size scaling law seems robust enough to appear here as well.

D Supplemental Figures

D.1 Average compute scaling

The scaling law presented in Fig. 4 is obtained by finding the Pareto optimal agents among all agents trained. It does not make clear whether the same scaling law would apply when training a single agent, rather than training with several random seeds and picking the best one. We therefore plot Fig. 8, which displays Elo scaling with compute, but with scores averaged over each neural-net size group. The scaling exponents deviate only slightly from the exponents calculated with individual agent data. This implies one can expect to find compute log-scaling of Elo also when training single agents, as opposed to training many seeds and picking the best performing one.

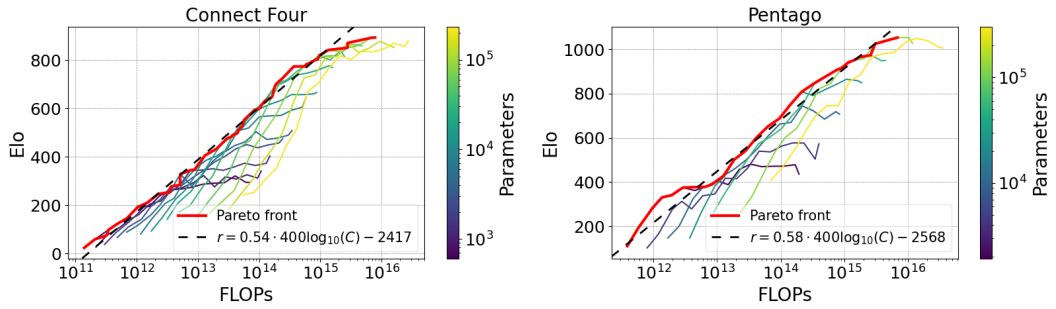

Figure 8: Averaged performance scaling with training compute. Re-plotting figure 4 with data averaged over 6 different seeds for each number of neural-net parameters produces a Pareto front with a similar scaling exponent.

D.2 Size scaling convergence

The parameter-count scaling law presented in section 5.1 only holds in the limit of infinite compute, when agents are allowed to train to convergence. A way of measuring the proximity to this limit is by looking at the evolution of the log-linear fit to the data. We have shown in Fig 3 that log-linear scaling appears already at short training lengths, but with lower slopes. In Fig 9 we plot the exponent values derived from these slopes. One can see that the exponent value slowly converges to its final value at 10^4 training steps.

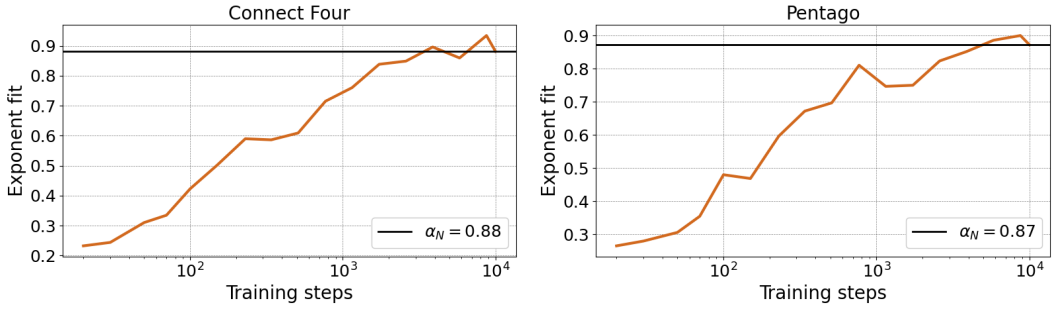

Figure 9: Convergence of the scaling exponent α_N. Fitting the curves in Fig 3 we obtain intermediate values for the scaling exponent during training. The exponent fits grow larger with training length, slowing down towards a final value for both games.

E Oware scaling

We measure the scaling laws of another popular board game, Oware. We choose this game because of its length; unlike Connect Four and Pentago, Oware games are not bound to end after a fixed number of turns. This also means the average number of turns per game is much larger compared to the other games, see Table 4. In practice, infinite games are handled in OpenSpiel by calling them off after $1{,}000$ steps have been played. These games, although rare, are excluded from our statistics. Another difference between Oware and the other games is the shape of the observation tensor. For Oware, neural network input is in the shape of a 14-elements vector of integers, counting the number of seeds in each pit. This is unlike the observation tensor of Connect Four and Pentago as well as games like Go, where board state is presented in binary encoding.

We find that Oware exhibits power-law scaling in playing strength in neural network size and compute, but encounter two serious issues during training. One issue is the lack of sufficient training time for agents to converge, not allowing us to gauge correctly the true value of the size scaling exponent. The other issue is that the largest models do not manage to reach their full potential, getting

Elo scores significantly lower than smaller models. We give a possible explanation to this in the last section. This issue mostly affects the correct estimation of the compute exponent.

Table 4: Game details including Oware. Oware games are not bounded to a maximal number of moves. As a result, they last much longer than Connect four or Pentago games on average. Additionaly, the neural network input comes in the form of integer numbers in Oware, rather than zeros and ones.

Game	Branching Factor	Average Game Length	Observation Shape
Connect Four	≤ 7	25.4 (max. 42)	$3 \times 6 \times 7$ Boolean tensor
Pentago	≤ 288	16.8 (max. 36)	$3 \times 6 \times 6$ Boolean tensor
Oware	≤ 6	100 (unbounded)	14-long integer vector

E.1 Size scaling

We repeat the analysis of neural network parameter scaling for Oware agents. Fig. 10 presents the scaling of Elo scores with model size. For Oware one observes, interestingly, that agents with large numbers of parameters do not train to optimality. Small- and medium-sized agents show however evidence of power-law scaling. It is clear that large agents could in theory reach better performance, since the range of strategies a MLP network could implement is always contained within that of a larger model. We speculate about the cause to this in section E.4.

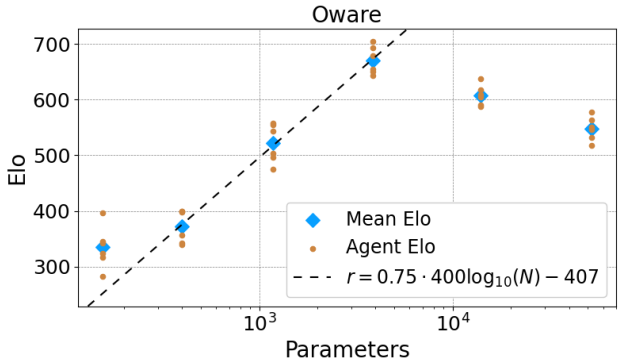

Figure 10: Oware size scaling. Oware shows a logarithmic scaling trend of Elo with neural net size that breaks abruptly for larger networks. Since the strategy space available to smaller agents is contained within that available to larger ones, large agents should at least reach an equal Elo score to smaller agents. It is therefore clear that the large agents did not converge to optimal performance.

Medium-sized agents performance follows a log scaling with model size, with a smaller scaling exponent than that found with Connect Four and Pentago. However when we plot the changing of the exponent fit during training in Fig. 11, it seems very likely that it will keep increasing with longer training. Unfortunately we could not extend training length by another order of magnitude in the time given for this analysis.

E.2 Compute scaling

Fig. 12 shows the Elo scores of Oware agents with different training compute. Despite the failure of large agents to learn effectively, Oware matches a log-linear fit over several orders of magnitude with a smaller exponent than that found in Connect Four and Pentago.

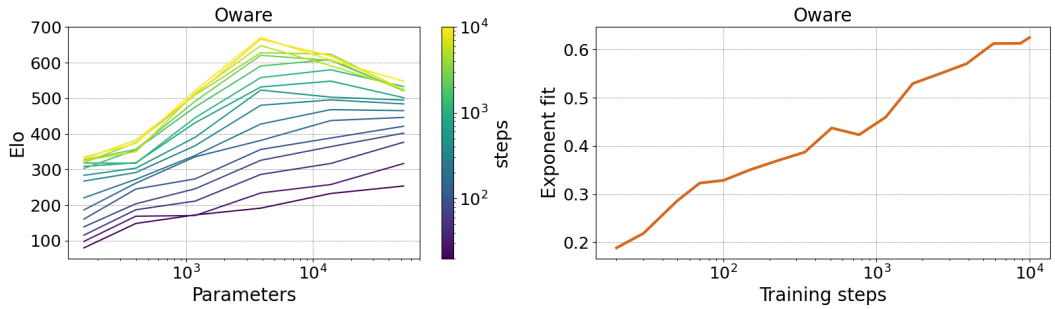

Figure 11: **Left:** Elo scaling with parameters during training. Log-linear scaling already appears at early stages of training, converging to a fixed curve. **Right:** α_N convergence. The exponent fit keeps increasing with training time for Oware, suggesting the final exponent should be larger.

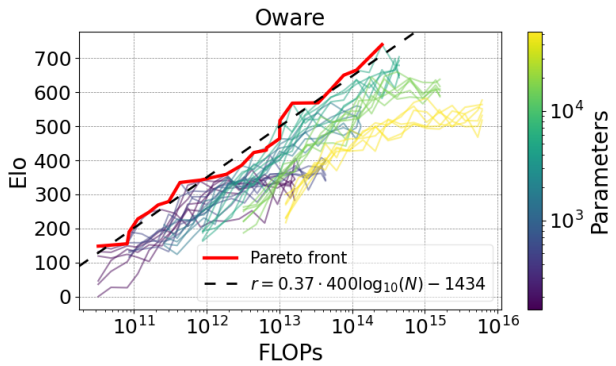

Figure 12: Oware performance scaling with training compute. The Pareto front follows a log-linear trend in compute. Large models, which failed to train well, do not make it to the Pareto front.

E.3 Estimating the compute optimal model size

We use the exponents α_N and α_C found for Oware to fit a new scaling law for the optimal model size, N_{opt}. Since Oware has a smaller compute-scaling exponent than the other games, its optimality exponent α_C^{opt} is also smaller. Overlaying the resulting power law on data from other games in Fig. 13, we find that it does not fit Connect Four and Pentago well. Even though the Oware exponent is significantly smaller, it still predicts that AlphaGo Zero and AlphaZero would benefit from an increase in model size.

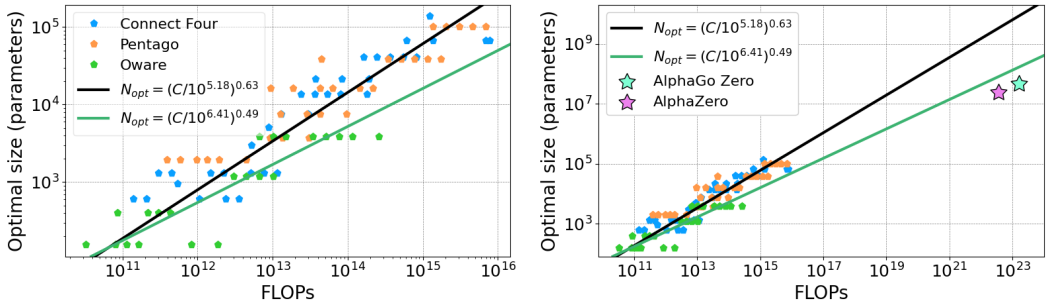

Figure 13: Optimal number of neural network parameters as in Fig. 1, with the scaling trend predicted with Oware in green. Even this lower exponent scaling law predicts previous state-of-the-art models were undersized, although to a much lesser degree.

E.4 TRAINING ISSUES

Reinforcement learning algorithms can exhibit convergence issues, as is evident in the case of Oware. The largest agents trained on this game clearly failed to converge to their optimal playing ability, since they under-performed smaller agents, which they should at least match in performance. We believe a likely cause to that is the format of the input tensor, which is composed of a vector of integers rather than a boolean tensor as used in Connect Four and Pentago and games like Go and chess (see Table 4). We observe the same trend of large models consistently failing to train well in another game, a toy model problem not presented here with floating-point inputs rather than binary ones. We find that large models trained on the toy problem with no issues when we use a modified version of AlphaZero created by us, which imports a key feature from AlphaGo Zero. Specifically, the modified version tests each new candidate for the data generating model against the current model, and only updates it if the new candidate is proved to perform better. OpenSpiel currently does not contain this feature, but we believe it could be useful in the case of non-boolean input.

Table 5: Scaling exponent for all three games. Oware shows smaller size and compute exponents, and a larger optimal size scaling exponent.

Exponents	Connect Four	Pentago	Oware
α_N	0.88	0.87	0.75
α_C	0.55	0.55	0.37
α_C^{opt}	0.62	0.63	0.49

F TEST SET LOSS

Most language model scaling laws revolve around scaling test-set loss as a power law of some quantity. In contrast, scaling laws for AlphaZero appear in the form of playing strength power-laws. When we look at the predictive ability of trained agents on a test dataset, we find that test loss does not scale as a power law.

To measure test loss scaling, we create a dataset of game states from random Connect Four games generated by taking random moves. We then annotate these states with a perfect solver (Pons, 2015). The solver provides us with the ground-truth state value ($1, 0, -1$ for win, draw loss from the current position, respectively) and a policy prior which is a uniform probability distribution over all optimal moves. Optimal moves are defined as those which lead to either the fastest victory, a draw otherwise, or the slowest loss otherwise. All sub-optimal moves are given probability zero.

We then compare the value estimations and policy priors created by the solver to those predicted by fully trained Connect Four agents, by calculating the value and policy losses. These are defined in the full AlphaZero loss function (Silver et al., 2017b):

$$l = (z - v)^2 - \boldsymbol{\pi}^\top \log \boldsymbol{p} + c||\theta||^2 \,, \tag{10}$$

where z is the recorded game outcome from the current position and $\boldsymbol{\pi}$ is the improved policy generated by the MCTS using the visit count of each node. We replace z and $\boldsymbol{\pi}$ with predictions made by the solver.

The resulting scaling trends appear in Fig. 14. For the policy head, we plot the test loss minus the irreducible loss (equal 0.336), the minimum loss achieved by matching the solver's policy prior exactly.

While policy loss does seem to decrease with model size, it is contained to a small range far from the minimal possible loss. Value loss decreases and then increases with model size. Both quantities do not seem to be good indicators of performance scaling.

We note that even though Elo score turns out to be a better measure of performance than test loss, there may exist even better measures. Elo rating fails to correctly model player dynamics when they are non-transitive, for example (Omidshafiei et al., 2019). Some alternatives exist (Omidshafiei et al., 2019; Oliveira et al., 2018), and it is possible they could capture elements of the agent dynamics that Elo rating would miss.

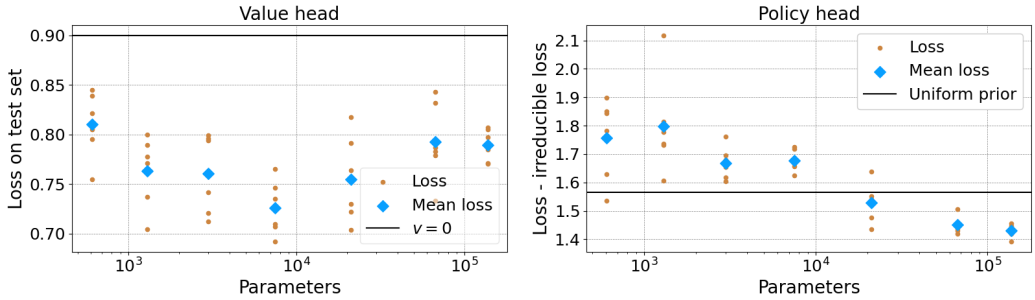

Figure 14: Test loss of Connect Four agents on a dataset created by a perfect solver. **Left:** Value head test loss. Loss changes very little with model size, large models have similar loss to small models. For comparison, we add a baseline loss received when always predicting a draw ($v = 0$). **Right:** Policy head test loss minus irreducible loss. Loss does decrease with size, indicating larger models produce a policy prior that is closer to the perfect policy. However the distance to the perfect prior is still large across all scales. For comparison, we add a baseline achieved when always predicting a uniform prior over all legal moves. Smaller agents achieve worse loss than the uniform baseline.