| Game | Board size | | Ordered board | | | | Permuted board | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Attention NN | | Convolutional NN | | Attention NN | | Convolutional NN | |
| | | | vs. 1× | vs. 10× | vs. 1× | vs. 10× | vs. 1× | vs. 10× | vs. 1× | vs. 10× |
| Breakthrough | 36 | $(6 \times 6)$ | 99%±1.1 | 94%±2.4 | 99%±0.8 | **98**%±1.4 | **99**%±0.7 | 93%±2.5 | 97%±1.7 | 93%±2.5 |
| Breakthrough | 64 | $(8 \times 8)$ | 83%±3.7 | 45%±4.9 | **88**%±3.2 | **65**%±4.7 | **85**%±3.5 | **53**%±4.9 | 77%±4.1 | 42%±4.8 |
| Breakthrough | 100 | $(10 \times 10)$ | 51%±4.9 | 9%±2.8 | 47%±4.9 | **15**%±3.5 | **17**%±3.7 | **4**%±1.9 | 4%±2.0 | 1%±0.8 |
| Connect Four | 42 | $(7 \times 6)$ | **72**%±4.3 | 57%±4.8 | 58%±4.8 | 53%±4.8 | 44%±4.8 | 37%±4.6 | **49**%±4.9 | **42**%±4.8 |
| English Draughts | 32 | $(8 \times 8)$ | 86%±2.3 | 43%±3.3 | 85%±2.5 | 46%±3.4 | 77%±2.8 | 36%±3.1 | **80**%±2.7 | **43**%±3.3 |
| Fox and Hounds | 32 | $(8 \times 8)$ | 91%±2.8 | **77**%±4.1 | 93%±2.5 | 72%±4.4 | 87%±3.3 | 58%±4.8 | **94**%±2.4 | **72**%±4.4 |
| Gomoku | 225 | $(15 \times 15)$ | **88**%±3.2 | 38%±4.8 | 44%±4.9 | 39%±4.8 | **74**%±4.3 | **14**%±3.4 | 10%±2.9 | 7%±2.5 |
| Hex | 25 | $(5 \times 5)$ | 83%±3.7 | 56%±4.9 | 85%±3.5 | 60%±4.8 | 79%±4.0 | 62%±4.8 | **86**%±3.4 | 65%±4.7 |
| Hex | 49 | $(7 \times 7)$ | **76**%±4.2 | **49**%±4.9 | 44%±4.9 | 22%±4.1 | **57**%±4.9 | **34**%±4.7 | 19%±3.9 | 8%±2.7 |
| Hex | 81 | $(9 \times 9)$ | **10**%±2.9 | 0%±0.5 | 2%±1.3 | 0% | **4**%±1.8 | 0%±0.5 | 0%±0.5 | 0% |
| Pentago | 36 | $(6 \times 6)$ | 82%±3.8 | 41%±4.8 | **94**%±2.4 | **72**%±4.4 | 70%±4.5 | 28%±4.4 | **80**%±4.0 | **46**%±4.9 |
| Reversi | 64 | $(8 \times 8)$ | 88%±3.2 | 68%±4.5 | 88%±3.1 | 71%±4.4 | **84**%±3.5 | **64**%±4.6 | 63%±4.7 | 35%±4.6 |
| The Mill Game | 24 | $(8 + 8 + 8)$ | 67%±4.3 | 36%±4.1 | 70%±4.2 | 40%±4.3 | 61%±3.9 | 34%±3.6 | 63%±4.4 | 34%±4.0 |
| Total average | | | 75% | 47% | 69% | 50% | 64% | 39% | 56% | 38% |

Table 1: The results of the NN agents with 600 iterations. The baseline opponent is UCT with 600 (1×) and 6,000 (10×) iterations. The dataset for each game was gathered for at most 2h using 20 CPU cores. The 95%-confidence intervals are given.

| Game | Size of dataset (MCTS plays) | | |
|---|---|---|---|
| | 200 | 400 | 1,000 |
| | **Generation time + Training time** | | |
| Breakthrough $(6 \times 6)$ | 6s + 20s | 11s + 20s | 25s + 20s |
| English Draughts | 1.1m + 20s | 1.3m + 20s | 3.5m + 40s |
| Reversi | 1.2m + 20s | 1.2m + 20s | 3.5m + 40s |
| | **Attention NN** | | |
| Breakthrough $(6 \times 6)$ | 4%±1.8 | 60%±4.8 | 74%±4.3 |
| English Draughts | 31%±2.9 | 42%±3.2 | 54%±3.3 |
| Reversi | 25%±4.2 | 22%±4.0 | 28%±4.3 |
| | **Convolutional NN** | | |
| Breakthrough $(6 \times 6)$ | 41%±4.8 | 48%±4.9 | 83%±3.7 |
| English Draughts | 42%±3.4 | 55%±3.5 | 60%±3.4 |
| Reversi | 6%±2.3 | 7%±2.4 | 21%±4.0 |

Table 2: Short training. The results of the NN agents with 600 iterations against UCT baseline with also 600 iterations.

paring it with CNNs based on random permutations of the input board tiles. Reordering the tiles should eliminate the spatially local correlation, which is assumed by default by CNNs.

**Fast model generation** To take advantage of the very fast RBG reasoner and to vastly decrease training time, the training dataset is generated by playing games using standard UCT MCTS agents (without NN). This approach provides better quality samples than in the early stages of self-play and allows gathering large amounts of data with limited resources. It can be easily parallelized since MCTS plays can be performed independently on CPU cores and do not require a GPU.

## Experiments with Conclusions

We evaluated the trained NN agents by playing 400 games (200 per side) against a standard MCTS agent with game tree reuse. The results of the constant simulations limit are presented in Table 1. Convolutional NN assumes that the board fits within a quad, and the tiles are given in order. To examine the robustness, we tested the behavior after obfuscating by permuting the tiles randomly (the same permutation used for both NNs). Overall, the attention NN has a similar performance to the CNN, even if the order of board tiles is random. The results vary a lot depending on the game, especially on its size. Yet, they suggest the trend that attention is a better choice for larger games, which is particularly visible in size variations of Breakthrough and Hex. It also suffers a smaller win ratio drop on average on our game set when the board is permuted (the mean drop is respectively 11% and 8% for attention NN vs. 13% and 12% for CNN).

Table 2 shows results after very short training. We used smaller NNs here, trained on much less data (less than 3% of the 2h dataset from Table 1). This experiment was inspired by (**?**), where in Breakthrough, after just 400 selfplay games, the winrate close to 80% was achieved. To achieve a similar playing strength, we needed about 1,000 MCTS vs. MCTS training plays. Thus, our plays are of worse quality, but we can generate them in less than a minute. Because of the RBG language efficiency (**?**), in our experiments, optimized Monte-Carlo playouts are much faster than NN evaluation. Such a situation, causing time-based comparison to favor pure MCTS, was not reported in GGP-related literature.

We conclude that knowledge of the action space and board topology is not required for obtaining good models. The data generated from MCTS plays in place of self-plays is still useful; in some cases, it allows beating the baseline after a few minutes of computing. Note that our research was focused on budget training, so the landscape could be different in longer settings.

## Acknowledgments

This work was supported in part by the National Science Centre, Poland under project number 2021/41/B/ST6/03691.

Maiores dolores itaque rerum consectetur, ea cumque fugiat.