Tiny Chess Networks Without Search

Strength against size from 5 k to 1 M weights

[GAP: author and affiliation]

[GAP: date]

Abstract

[GAP: Written after the freeze. It states: the question (how strong a chess network can be at a given number of stored weights when it plays one network pass per move, with no search); the sizes (about 5 k to 1 M weights, released at 19 k, 123 k and 979 k); the method in one sentence (distilling the policy of a large Lc0 network on positions from Lichess's evaluation file, with inputs computed from the rules, shared relative-position attention, pruning while training and looped blocks); the headline numbers on Stockfish's levels; the smallest size that still scores half the points against Stockfish's lowest level; and what is released.]

1 Introduction

Strong chess programs search. A network that plays the move its policy rates highest, after a single pass and with no look-ahead, is a different object: its whole playing strength has to be stored in its weights. Ruoss et al. [10] showed that a large transformer trained on engine labels reaches grandmaster level this way, and Leela Chess Zero's networks play strongly from the policy alone [8]. Both sit at tens to hundreds of millions of weights.

This paper asks the question at the other end of the scale. How much chess fits in a thousand times fewer weights? We train networks from about 5 k to about 1 M weights and measure each one's strength when it plays one pass per move. Our contribution is a measured curve of strength against size, the method that produced each point on it, and the list of changes that did not help.

The setting has two rules. First, *no search at play*: in every measured game the move is the legal move with the highest policy value from one forward pass. Second, *anything may be used before play*: data, engine labels and larger teacher networks are all allowed in training. We count *stored weights*, the parameters that are not zero. Two of our methods buy strength with computation instead of weights, and we say so where they appear: a pruned network computes like its larger parent unless sparse arithmetic is used, and a looped network runs the same block several times (Section 4.7).

Our findings, in the order of the paper:

- **The size curve.** [GAP: One sentence with the three release sizes' numbers and the slope: Elo per doubling of the weights, on the dense line.]
- **What moved the curve.** At 123 k weights the largest single gains came from the learning rate, the training length, the data, inputs computed from the rules, and reusing weights by looping; Section 4 gives each with its measured gain.
- **How small.** A network of 19 k weights scores more than half the points against Stockfish's lowest level. [GAP: Where the bar is lost: results of the 14 k, 10 k and 5 k runs.]
- **What did not help.** Section 6 lists every change we tested and left out, with its number.

- **Cost.** Everything was trained on one GTX 1080 Ti and on rented consumer cards; Section 7 gives the hours and the money.

2 Related work

Chess without search. Ruoss et al. [10] train transformers of up to 270 M parameters on Stockfish action values and report play at grandmaster level with no search; their smallest network has 9 M parameters [CHECK: sizes and the reported ratings]. Lc0's transformer networks are described by Monroe and the Leela Chess Zero team [8]; we use one of them, BT4, as the teacher. Maia [7] trains networks to predict human moves at a given rating and plays them without search. [GAP: The small searchless networks published by hobbyists that we measured on our own ladder, with their sizes and how we cite them.]

Small evaluation networks. NNUE [9] made small networks standard inside searching engines; there the network is an evaluation function called millions of times, a different regime from ours.

Distillation and pruning. We distil a teacher's policy [4]. Our pruning is gradual magnitude pruning during training [11]; that a sparse network trained from a larger dense one can beat a dense network of the same count is in line with the lottery-ticket line of work [3].

Reusing weights across depth. Running one block several times follows the Universal Transformer [2] and ALBERT [6]. [CHECK: whether a closer chess or game-playing precedent for looped blocks exists]

Lookup memory. Conditional memory by hashed lookup tables [1] trades stored parameters for computation, the opposite of the trade this paper studies; we did not test it.

3 Setting and measurement

3.1 The player

The network sees the current position only, from the side to move: twelve piece planes, an en-passant plane and a plane of ones, with the four castling rights, a repetition flag and the fifty-move counter as six single numbers. There is no move history. The policy is a list of 1,858 moves as in Lc0, and illegal moves are masked. At play the network makes one forward pass and plays the legal move with the highest value.

3.2 Strength on Stockfish's levels

Every comparison in this paper is played against Stockfish 16 with its strength limit set to 1320, 1500, 1700 and 1900, at 40,000 nodes a move. A network plays 500 games against each level from a fixed set of openings, each opening with both colours, and the result is fitted to one Elo number by solving $\sum_\ell (p_\ell - n\,E_\ell(x)) = 0$ with $E_\ell(x) = 1/(1 + 10^{(R_\ell - x)/400})$, where p_ℓ is the points scored against level ℓ of rating R_ℓ and n the games played. We repeat the whole bench on three seeds and report the mean of the three fits.

The 17 Elo rule. [GAP: How the bar was measured: the spread of repeated benches of one network.] A change counts as a gain only if it moves the three-seed number by at least 17 Elo. Differences under that are reported as "inside the noise".

Screens. Before a full run a change is tried in a short run of 30,000 steps on 10 M positions; it passes if its held-out policy loss is at least 0.005 below the baseline's. A pass earns a full run, and only the full run's bench decides. Several changes passed the screen and then showed nothing in games (Section 6).

Stronger networks. For networks near and above 1,900 we add three higher levels. [GAP: The three levels and the seven-level fit.]

3.3 Other scales

The Stockfish-level number is our own scale and is not a rating anyone else publishes. For readers who want a familiar number we give two more, both for display only. [GAP: The bot scale: a round robin among our networks and published searchless networks, fitted with BayesElo and pinned at a published rating. The people scale: estimates against published Lichess ratings of the same outside networks. Which of the three the main figure uses.]

3.4 The smallest-network bar

We call a network "still playing chess" if it scores at least half the points against Stockfish's lowest level (1320) on each of the three seeds, 500 games a seed.

4 Method

4.1 Data and teacher

The training positions are the 104 M positions of Lichess's evaluation file. [CHECK: exact file, date and count; the held-out positions removed from training] The target for each position is the whole policy of Lc0's BT4 network (about 185 M weights) at temperature 2. There is no value target: adding one cost 30.5 Elo at 123 k. [GAP: How the teacher's policies were computed and stored.]

4.2 Network

The body is a transformer over the 64 squares [CHECK: pre-norm or post-norm, activation], with a feed-forward layer four times the width and Lc0's attention policy head, in which each move is scored from its from-square's query and its to-square's key. The dense 123 k network has width 32, 8 blocks and 8 heads.

4.3 Training

Muon [5] on the blocks' matrices and AdamW on the rest, one peak rate for both (1.6e-2 at 123 k and above, 3.2e-2 for the smallest networks), 2,000 steps of warm-up and a cosine to 1e-5, weight decay 0.01, batches of 1,024, and 234,000 steps. Table 1 gives what each choice was worth at 123 k.

4.4 Inputs computed from the rules

Five groups of extra input planes are computed from the position by the rules alone, with no evaluation in them: which squares each side attacks, pins, legal-move counts, piece counts and material. [CHECK: the exact planes of each group and their number] They cost no stored weights beyond a wider input layer. [GAP: Their measured gain on the dense line at 123 k and at 979 k.] Four more groups (attacks by piece type, hanging pieces, pawn structure, mobility) were screened and did not earn a place.

Table 1: Changes that joined the recipe at about 123 k weights, in the order they were made. Each row is read against the row above it on the Stockfish levels. Rows marked † were read on one bench of an earlier ladder and are shown for the order only.

Change	Gain (Elo)
Lichess-evaluation positions in place of elite human games†	+80
Muon in place of AdamW†	+31
No value target	+30.5
Peak learning rate 1.6e-2 in place of 1e-3	+57.5
234,000 steps in place of 117,000	+34.1
8 attention heads in place of 4	+37.1
Dense 123 k network after these	1,706.2

4.5 Relative positions

Each attention head adds a learned bias that depends on the two squares' relative position, with one table shared by all blocks. [CHECK: the exact form] At 123 k it adds [GAP: dense gain] on its own; with the rule inputs on a pruned network the two together add 32.9 Elo over the plain pruned network. At 19 k it adds nothing (−2.4).

4.6 Pruning while training

A parent network about 2.5 to 3 times the target size is trained with the same recipe, and its smallest weights are set to zero on a schedule from step 4,000 to step 150,000, every 2,000 steps, until the target count remains. At one stored-weight count the pruned network beats the dense one at every size we read: +30.6 Elo at 123 k, +28.3 at 311 k and +26.0 at 979 k. A pruned network computes like its parent unless sparse arithmetic is used; we report it in its own section for that reason, and the paper's main line is dense.

4.7 Looping

A looped network runs its stack of blocks several times. The input is added again before each later pass together with one learned vector per pass. At 123 k, one block of width 90 looped eight times (124,267 weights) reaches 1,764.4 against 1,724.0 for the plain network with the same inputs, a gain of 40.4 Elo, and matches the pruned network (1,769.7) while staying dense. [GAP: The looped 979 k: the shape screen and the full run. The pruned looped 123 k.] Looping lost at 48 k in a short screen, so the smallest networks are not looped.

What looping costs. A network looped n times does the arithmetic of a plain network with n times its weights. Our claim for looped networks is strength per stored weight, not per unit of computation. [GAP: Measured time per position on one CPU thread for the plain and looped networks at each size.]

5 Results

[GAP: Main figure: strength against stored weights, dense line and pruned line, with the looped points marked, from 5 k to 979 k.]

5.1 The size curve

[GAP: Elo per doubling on the dense line; where the curve bends.]

Table 2: Strength against size. Provisional until the recipe freeze: every number here is filled from the findings record at the freeze.

Size	Network	Stored weights	Stockfish levels
5 k	dense / pruned	[GAP:]	[GAP:]
10 k	dense / pruned	[GAP:]	[GAP:]
14 k	dense / pruned	[GAP:]	[GAP:]
19 k	dense / pruned	[GAP:]	[GAP:]
41 k	dense	[GAP:]	[GAP:]
123 k	dense / looped / pruned	[GAP:]	[GAP:]
311 k	dense / pruned	[GAP:]	[GAP:]
979 k	dense / looped / pruned	[GAP:]	[GAP:]

5.2 How small

[GAP: The bar against Stockfish's lowest level at 19 k, 14 k, 10 k and 5 k, dense and pruned, three seeds each; between which sizes it is lost.]

5.3 Training length at 979 k

At 123 k, doubling the steps again from 234,000 to 468,000 added 3.2 Elo. At 979 k the same doubling added 23.5 Elo on four levels (16.4 on seven), so the released 979 k network is trained for 468,000 steps.

5.4 Against other networks

[GAP: Our networks beside the published searchless networks on the bot scale and the people scale, with each outside network's size.]

6 What did not help

Each line was tested as Section 3 describes. "Inside the noise" means under 17 Elo on three seeds.

- **A value target** beside the policy: −30.5 Elo at 123 k.
- **Teacher temperature** 1 or 3 in place of 2: −24 and −44 Elo on early single benches.
- **Mixing in the deep best move** of the Lichess file for a fifth of the target: +1.9.
- **Look-ahead targets** from a short search, for a whole run or a last stage: −5 to −51 Elo.
- **A second stage on the student's own games**, labelled by the teacher: +2.5.
- **More data at fixed length**: 104 M positions in place of 10 M read −6.2 and +7.2 at the two lengths; we kept the larger set.
- **Smolgen**, Lc0's position-dependent attention bias: no gain at equal weights at 979 k in a screen, and a tenth slower.
- **The attack graph**, an attention bias from which pieces attack which squares: it passed its screen at 48 k and then added 2.0 Elo at 19 k at full length, at about 61% of the training speed.

- **Four more groups of rule inputs**: screened at 123 k and 48 k, the best a narrow pass that was not carried on.
- **Relative positions at 19 k**: −2.4.
- **Looping at 48 k**: lost its screen.
- **A larger pruning parent** at 979 k: [GAP: the two larger parents' numbers].
- [GAP: The remaining rows of the findings' "tried and left out" table: shapes, head counts, optimiser settings, the second-stage mixes.]

One pattern is worth stating. A screen win of 0.005 to 0.01 in held-out loss did not show in games in our tests, while wins of 0.03 did: the looped 123 k network gained 0.029 in its screen and 40 Elo at full length.

7 Costs

All home training ran on one GTX 1080 Ti. [GAP: Hours for each release network.] Rented runs used consumer cards at $0.16 to $0.34 an hour: [GAP: the money per release network and the total, including runs lost to host faults.] [GAP: Games played for the benches and the CPU hours.]

8 Limitations

- One training seed per network. The 17 Elo rule covers game noise and bench seeds, not the spread between training runs.
- Our strength numbers are against one engine family at fixed settings on one machine; the other two scales are estimates.
- Pruned and looped networks are smaller in storage, not in computation.
- The networks see no move history, so they cannot tell a repeated position from a new one beyond the single repetition flag.
- [GAP: Anything the freeze adds.]

9 Conclusion

[GAP: Written after the freeze.]

Release

[GAP: What is released: the weights of the 19 k, 123 k and 979 k networks, the code, and the page where they can be played.]

References

[1] Xin Cheng et al. Conditional memory via scalable lookup: A new axis of sparsity for large language models. *arXiv preprint arXiv:2601.07372*, 2026.

[2] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019.

[3] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

[4] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[5] Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. Check the canonical reference and URL.

[6] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

[7] Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Aligning superhuman AI with human behavior: Chess as a model system. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

[8] Daniel Monroe and The Leela Chess Zero Team. Mastering chess with a transformer model. *arXiv preprint arXiv:2409.12272*, 2024.

[9] Yu Nasu. Efficiently updatable neural-network-based evaluation functions for computer shogi, 2018. The 28th World Computer Shogi Championship Appeal Document.

[10] Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Li Kevin Wenliang, Elliot Catt, John Reid, and Tim Genewein. Grandmaster-level chess without search. *arXiv preprint arXiv:2402.04494*, 2024.

[11] Michael Zhu and Suyog Gupta. To prune, or not to prune: Exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.