Stockfish or Leela Chess Zero? Comparing Performance Against Endgame Tablebases

Asmaul Husna
CCID: asmaul

Quazi Asif Sadmine
CCID: sadmine

Project

Vs

Goals of Our Experiment

- How these two leading chess engines respond against perfect play?
- Which is easier to predict for the engines? Wins or draws?
- How well they perform at different search budgets?
- Is there any specific pattern where one of the engines make more mistakes than the other?

Stockfish

- **Search algorithm:** Alpha-beta pruning
- **Neural Network:** NNUE

Lc0

- **Search algorithm:** MCTS
- **Neural Network:** Deep Convolutional Neural Network

Experimental Setup

- Stockfish version 15.1, available at https://stockfishchess.org
  - Backend: CPU
  - Network with 2850 Elo rating
- Lc0 version 0.29.0, available at https://lczero.org
  - Backend: opencl
  - Network with 2850 Elo rating
- 3 pieces endgames:
  - KQk, KRk, KBk, KNk
- 4 pieces endgames:
  - KPkp, KRkr, KRkp, KQkr, KQkq, KQkb, KQkp
- 5 pieces endgames:
  - KQRkq
- Endgame tablebases:
  - Syzygy
  - Gaviota

Some Experimental Results

Total number of mistakes for 3 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KQk	20977	19 (0.09%)	173 (0.8%)
KRk	24755	0	23 (0.09%)
KNk	53806	0	0
KBk	52234	0	0

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KPkp	743609	13541 (1.82%)	9437 (1.3%)
KRkr_win	784918	12759 (1.63%)	1829 (0.23%)
KRkr_draw	1892778	16313 (0.9%)	7309 (0.4%)
KRkp_win	1110806	39490 (3.56%)	13032 (1.17%)
KRkp_draw	398282	16417 (4.12%)	12486 (3.13%)
KQkr_win	890800	8512 (0.96%)	7062 (0.8%)
KQkr_draw	49184	4745 (9.6%)	2261 (4.6%)

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KPkp	743609	13541 (1.82%)	9437 (1.3%)
KRkr_win	784918	12759 (1.63%)	1829 (0.23%)
KRkr_draw	1892778	16313 (0.9%)	7309 (0.4%)
KRkp_win	1110806	39490 (3.56%)	13032 (1.17%)
KRkp_draw	398282	16417 (4.12%)	12486 (3.13%)
KQkr_win	890800	8512 (0.96%)	7062 (0.8%)
KQkr_draw	49184	**4745 (9.6%)**	**2261 (4.6%)**

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KPkp	743609	13541 (1.82%)	9437 (1.3%)
KRkr_win	784918	12759 (1.63%)	1829 (0.23%)
KRkr_draw	1892778	16313 (0.9%)	7309 (0.4%)
KRkp_win	1110806	39490 (3.56%)	13032 (1.17%)
KRkp_draw	398282	**16417 (4.12%)**	**12486 (3.13%)**
KQkr_win	890800	8512 (0.96%)	7062 (0.8%)
KQkr_draw	49184	4745 (9.6%)	2261 (4.6%)

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KPkp	743609	13541 (1.82%)	9437 (1.3%)
KRkr_win	784918	12759 (1.63%)	1829 (0.23%)
KRkr_draw	1892778	16313 (0.9%)	7309 (0.4%)
KRkp_win	1110806	**39490 (3.56%)**	**13032 (1.17%)**
KRkp_draw	398282	**16417 (4.12%)**	**12486 (3.13%)**
KQkr_win	890800	8512 (0.96%)	7062 (0.8%)
KQkr_draw	49184	4745 (9.6%)	2261 (4.6%)

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KPkp	743609	**13541 (1.82%)**	**9437 (1.3%)**
KRkr_win	784918	12759 (1.63%)	1829 (0.23%)
KRkr_draw	1892778	16313 (0.9%)	7309 (0.4%)
KRkp_win	1110806	**39490 (3.56%)**	**13032 (1.17%)**
KRkp_draw	398282	**16417 (4.12%)**	**12486 (3.13%)**
KQkr_win	890800	8512 (0.96%)	7062 (0.8%)
KQkr_draw	49184	4745 (9.6%)	2261 (4.6%)

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KQkq_win	934428	18038 (1.93%)	14693 (1.57%)
KQkq_draw	1293823	8874 (0.7%)	6724 (0.52%)
KQkp_win	945359	**4359 (0.46%)**	**7751 (0.82%)**
KQkp_draw	155352	2284 (1.47%)	1454 (0.94)
KQkb_win	701738	787 (0.11%)	2633 (0.38%)
KQkb_draw	220956	843 (0.38%)	538 (0.24%)

Some Experimental Results

Total number of mistakes for 4 pieces

EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KQkq_win	934428	18038 (1.93%)	14693 (1.57%)
KQkq_draw	1293823	8874 (0.7%)	6724 (0.52%)
KQkp_win	945359	**4359 (0.46%)**	**7751 (0.82%)**
KQkp_draw	155352	2284 (1.47%)	1454 (0.94)
KQkb_win	701738	**787 (0.11%)**	**2633 (0.38%)**
KQkb_draw	220956	843 (0.38%)	538 (0.24%)

Some Experimental Results

Total number of mistakes for 5 pieces

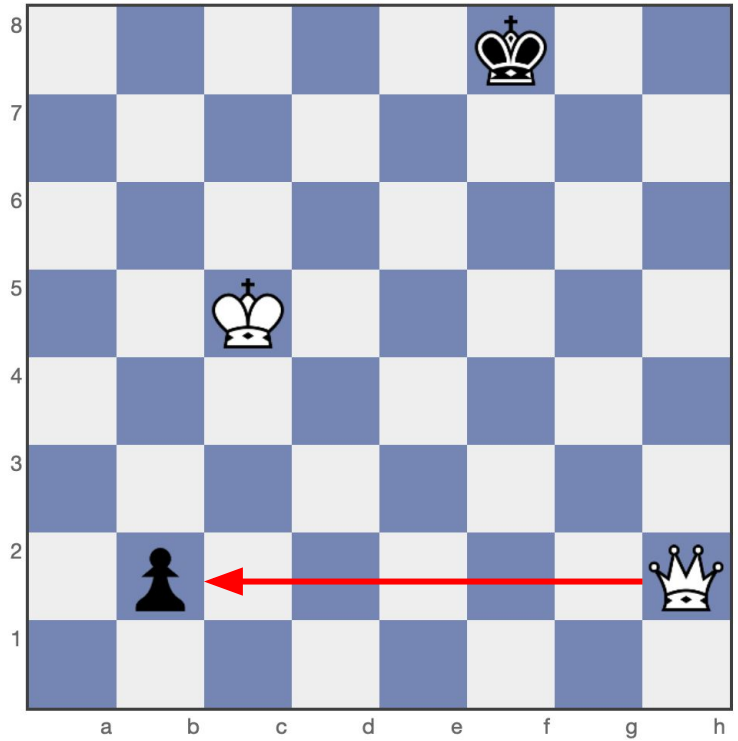
EGTB	Total Positions Tested	Stockfish (Policy)	Lc0 (Policy)
KQRkq_win	710730	26024 (3.66%)	44263 (6.23%)
KQRkq_draw	115308	22215 (19.3%)	6724 (5.83%)

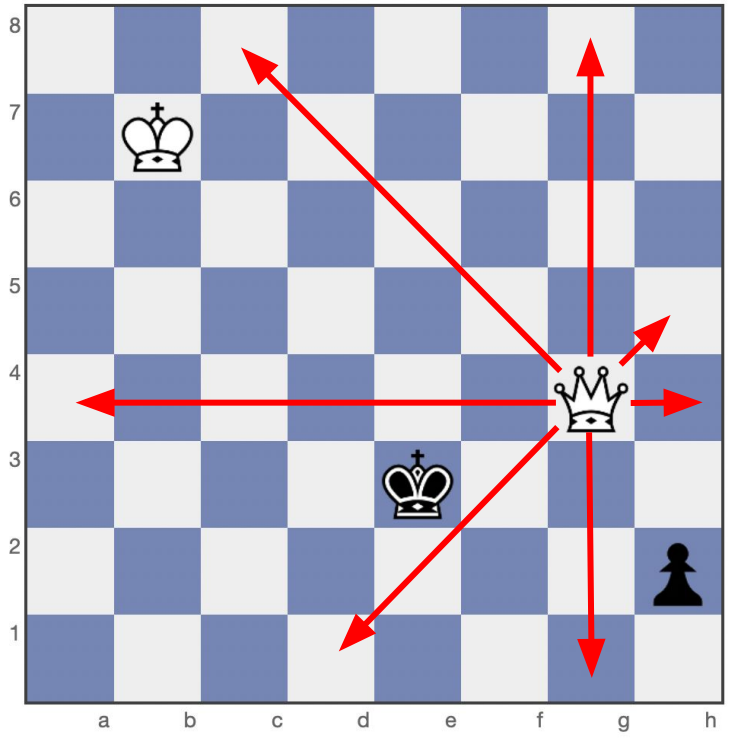
Some Experimental Results

Total number of mistakes

EGTB	Stockfish		Lc0	
	Policy	400	Policy	400
KRkr_win	1.63%	0.36%	0.23%	0
KRkr_draw	0.9%	0.11%	0.4%	0
KRkp_win	3.56%	0.64%	1.17%	0.06%
KRkp_draw	4.12%	0.71%	3.13%	0.08%

Pawn Position Analysis





Some Experimental Results

Pawn position analysis of **Stockfish**

EGTB	Mistakes in Pawn Attacked Position	Percentage	Mistakes in Pawn Safe Position	Percentage
KQkp_win	1536 (440955)	0.34	2823 (504404)	0.55
KQkp_draw	549 (56718)	0.96	1735 (98634)	1.76
KRkp_win	4139 (355035)	1.16	35351 (755771)	4.67
KRkp_draw	1085 (71592)	1.51	15332 (326690)	4.75
KPkp	1666 (97752)	1.70	11875 (645857)	1.84

Some Experimental Results

Pawn position analysis of **Leela Chess Zero**

EGTB	Mistakes in Pawn Attacked Position	Percentage	Mistakes in Pawn Safe Position	Percentage
KQkp_win	5001 (440955)	1.13	2750 (504404)	0.54
KQkp_draw	347 (56718)	0.61	1107 (98634)	1.12
KRkp_win	1161 (355035)	0.32	11871 (755771)	1.57
KRkp_draw	436 (71592)	0.60	12050 (326690)	3.69
KPkp	1256 (97752)	1.28	8181 (645857)	1.26

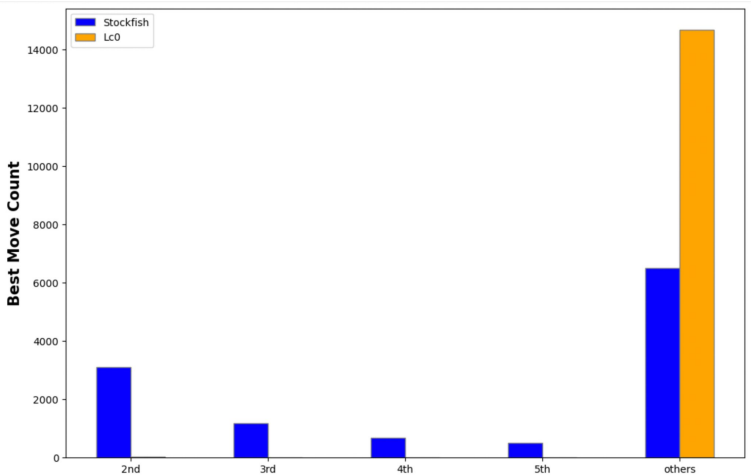
MultiPV Analysis

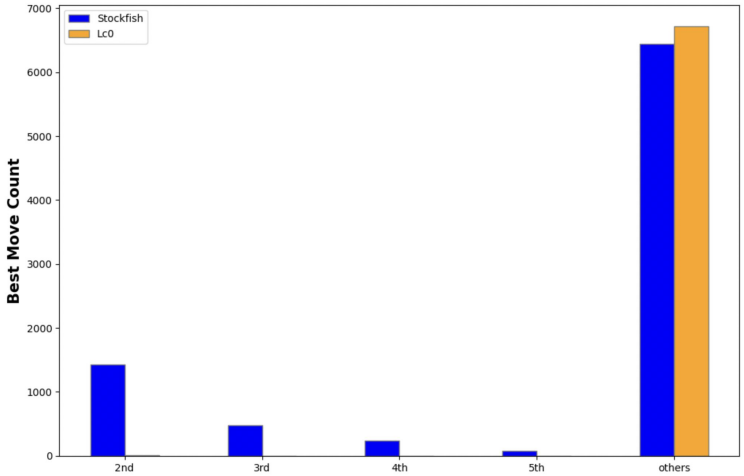
MultiPV:

- PV = Principal Variation
- PV is a sequence of moves that the engine considers as best
- MultiPV contains multiple sequences of moves
- The first move in each PV is the n-th best move (n = PV number) in the current board position

```
1: [Move.from_uci('g4e6'), Move.from_uci('e3d2')]
2: [Move.from_uci('g4g5'), Move.from_uci('e3e2')]
3: [Move.from_uci('g4h3'), Move.from_uci('e3d2')]
4: [Move.from_uci('g4g3'), Move.from_uci('e3d2'), Move.from_uci('g3h2'), Move.from_uci('d2c1')]
5: [Move.from_uci('g4g2')]
```

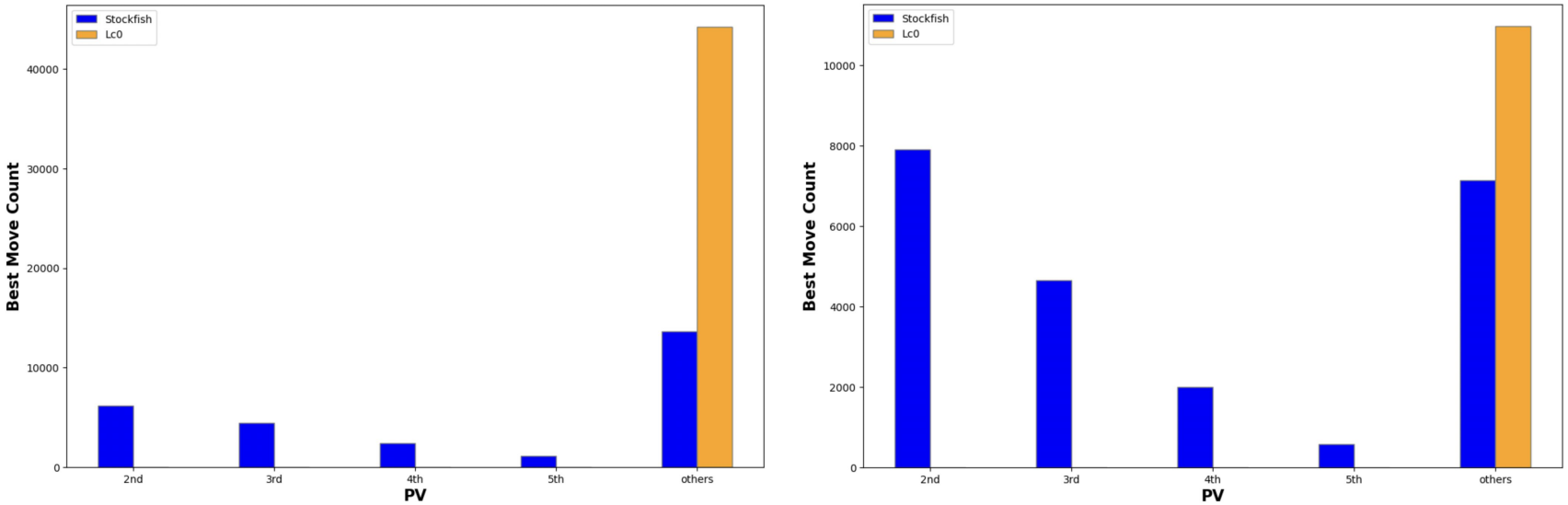
MultiPV Analysis for 4 pieces





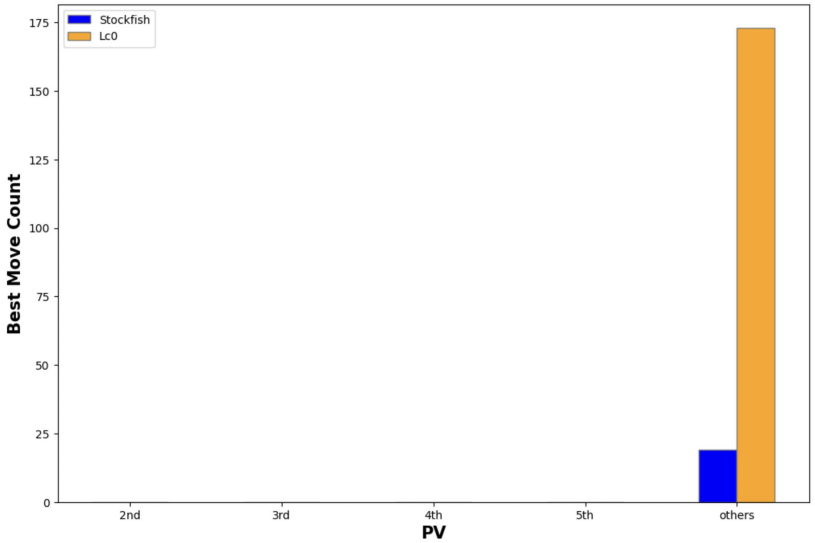
Analysis of top 5 moves for mistaken positions of KQkq_win (left) and KQkq_draw (right)

MultiPV Analysis for 5 pieces



Analysis of top 5 moves for mistaken positions of KQRkq_win (left) and KQRkq_draw (right)

MultiPV Analysis for 3 pieces



Analysis of top 5 moves for mistaken positions of KQk

Conclusion

- Stockfish policy is strictly better than or equal to Lc0 policy in 3 pieces endgames
- Lc0 policy is better than stockfish policy in most of the positions of 4 pieces endgames
- For 5 pieces endgames, no conclusion can be decided
- With search, stockfish starts performing much better
- Lc0 policy is nowhere in comparison with stockfish policy in case of time to compute results
- When the pawn is attacked by any stockfish piece, the engine policy makes less mistakes whereas, lc0 policy has no direct relationship with any attacked position of the pawn
- In a mistaken position, Lc0 policy usually does not have the best move in its top 5 moves
- In a mistaken position, stockfish policy can have more numbers of best move in its top 5 moves

Future Work

- Find more interesting engine behaviours in the mistaken positions and compare them
- Experiment with larger endgame pieces

Any Question?