ChessLab: Evaluating the Wisdom of Artificial Crowds in Chess

Anonymous Author(s)
Submission Id: «submission id»

ABSTRACT

Background: *Wisdom of crowds* suggests that aggregated decisions of diverse, independent agents often outperform individual experts. However, crowds of casual chess players still struggle against strong opponents. Recent projects show that the method by which a move is chosen by the crowd can significantly influence their performance.

Objectives and Research Questions: This work investigates whether artificial crowds can exhibit collective intelligence phenomena in chess. Key questions include: (1) Can voting ensembles of weak agents defeat stronger opponents? (2) How does diversity composition affect collective performance? (3) What voting mechanisms optimally aggregate engine decisions?

Methods: We introduce ChessLab, an open-source framework that replaces human crowds with heterogeneous ensembles of deterministic and learning-based chess engines. The system enables reproducible experiments with detailed move-level evaluations and robust Elo estimation across diverse agent types on a unified scale. We validate this methodology through Elo coherence experiments, establishing MadChess as a practical reference standard due to its wide Elo range and reliable self-consistency.

Results: Preliminary experiments show that majority voting among diverse weak engines does outperform significantly the mean level of those engines.

Conclusions: This paper presents the framework, experimental methodology, and a research roadmap bridging empirical chess results and broader theories of collective intelligence.

KEYWORDS

chess engines, wisdom of crowds

ACM Reference Format:
Anonymous Author(s). 2026. ChessLab: Evaluating the Wisdom of Artificial Crowds in Chess. In *Appears at the 8th Games, Agents, and Incentives Workshop (GAIW-26). Held as part of the Workshops at the 25th International Conference on Autonomous Agents and Multiagent Systems., Paphos, Cyprus, May 2026*, IFAAMAS, 9 pages.

1 INTRODUCTION

Collective intelligence can in some circumstances exceed individual capability. One of the earliest demonstrations comes from Galton's famous ox-weighing experiment, where the median guess of approximately 787 individuals in a county fair was more accurate than expert estimators and nearly matched the actual weight of 1,197 pounds [11]. This phenomenon has been tested in a variety of settings, such as prediction markets, where Atanasov et al. compared prediction markets with prediction polls in a large geopolitical forecasting tournament and found that carefully aggregated team forecasts could outperform market prices [2]. Question-answering game shows also provide natural experiments. In "Who Wants to Be a Millionaire", the "Ask the Audience" lifeline yields correct answers around 87–92% of the time, whereas phone-a-friend experts succeed only about 55–65% of the time, across many national editions of the show [27].

However, in competitive games like chess, crowds have historically struggled against strong opponents. The 1999 Kasparov vs. The World match saw more than 50,000 human players collectively challenge the World Chess Champion Garry Kasparov [14], reaching move 62 before conceding. More recently, in May 2025, over 143,000 players held World Champion Magnus Carlsen to a draw in a record-breaking game on Chess.com [7], and in November 2025, over 200,000 players lost in a match against International Master Levy Rozman (GothamChess) [6]. However, these matches remain largely anecdotal: a single game provides minimal evidence for systematic collective intelligence, as outcome variance is high and learning effects are confounded.

A Fouloscopie project, led by Mehdi Moussaïd at the Max Planck Institute for Human Development, addressed this limitation by conducting one of the first rigorous, large-scale experiments on collective chess intelligence [20]. Approximately 25,000 participants were split into teams to play a total of 500 chess games against AI opponents. Using majority-vote aggregation of player moves, Moussaïd and colleagues demonstrated that crowds achieved a performance level similar to a master while being composed predominantly of beginners and casual players. Importantly, by controlling for poll visibility, the results showed a significant effect of social influence on the crowd's level.

Yet even this landmark study leaves open critical questions:

- **Mechanism understanding**: What aspects of diversity (in skill, size, architecture) drive the ensemble advantage?
- **Optimization**: Can we identify decision-making processes that outperform simple majority aggregation?
- **Domain generality**: Does collective intelligence in chess rely on human-specific cognitive mechanisms, or do the principles apply to artificial agents as well?

We address these questions through **ChessLab**, an open-source framework for studying artificial collective intelligence in chess. By substituting human crowds with heterogeneous ensembles of chess engines, we gain precise control over experimental conditions, perfect reproducibility, and the ability to run experiments rapidly with modest computational cost.

The main contributions of this paper are:

(1) **Framework and methodology**: A modular, extensible system for running ensemble chess experiments at scale, with

Appears at the 8th Games, Agents, and Incentives Workshop (GAIW-26). Held as part of the Workshops at the 25th International Conference on Autonomous Agents and Multiagent Systems., Armstrong, Curry, Hosseini, Mattei, Tsang, Wąs (Chairs), May 2026, Paphos, Cyprus. © 2026 Copyright held by the owner/author(s).

persistent storage of games, moves, evaluations, and statistical analyses.
(2) **Theoretical grounding**: Clear hypotheses derived from Scott Page's Diversity Prediction Theorem and classical wisdom-of-crowds theory, adapted to the chess domain.
(3) **Comprehensive research roadmap**: A structured plan for experiments addressing collective intelligence across different engine types, ensemble compositions, and voting strategies.

This paper focuses on framework description, experimental design, and hypothesis articulation. Results from additional experiments will be presented in subsequent publications as they are completed.

2 RELATED WORK

2.1 Collective Intelligence and Wisdom of Crowds

The wisdom of crowds phenomenon emerged from early empirical observations: Galton's ox-weighing experiment demonstrated that collective judgments (median of 787 guesses) were more accurate than nearly all individual estimates [11]. Surowiecki systematized this observation, arguing that under certain conditions (diversity of opinion, independence of members, decentralization, and effective aggregation) groups reliably outperform experts [27]. Formally, Scott Page's *Diversity Prediction Theorem* provides mathematical grounding:

$$\text{Collective Error} = \overline{\text{Individual Error}} - \text{Diversity} \qquad (1)$$

where collective error is the mean squared error of the group prediction, $\overline{\text{Individual Error}}$ is the average of individual squared errors, and Diversity captures how different agents' predictions are [22]. This result implies that a team of ordinary individuals can outperform a specialist, provided the group is large enough and exhibits sufficient diversity in cognitive approach.

However, diversity alone is insufficient. Studies show that social influence, information cascades, and herding can destroy the diversity necessary for wisdom of crowds [1, 27]. Almaatouq et al. demonstrated that agents strategically choosing whom to learn from (adaptive networks) better maintain collective intelligence than static networks [1]. Koriat and colleagues showed that *confidence* acts as a metacognitive signal: when individuals can estimate how well-calibrated their judgment is, confidence-weighted aggregation outperforms simple majority voting [16], except when facing common misconceptions that result in misplaced confidence. Prelec extended this with the "surprisingly popular" voting method, which selects answers that are more popular than subjects predicted, revealing hidden consensus even when the majority is wrong [23].

In summary, collective intelligence requires: (1) a sufficient number of participants, (2) diversity in decision-making, (3) independence of judgments, (4) appropriate aggregation mechanisms, and possibly (5) calibration of confidence when weighting is available.

2.2 Collective Intelligence in Chess

Chess provides an ideal laboratory for studying collective decision-making in complex, sequential tasks. Unlike static judgment tasks (trivia answering, estimation), chess involves:

- **High-dimensional state space**: Approximately 10^{47} legal positions, making expert heuristics difficult and providing an extensive test dataset.
- **Strategic thinking**: Each player must evaluate not just the next move but plan for responses and counter-responses. This task allows deliberation to express emerging strategies.
- **Objective evaluation**: The outcome of a game is unambiguous, and thanks to modern engines, each move can be evaluated with precise numerical scoring. The performance level of a crowd can thus be estimated with high precision.
- **Cultural significance**: Chess remains a synecdoche for intelligence. Both humans and algorithms are expected to demonstrate competence at this task, providing abundant documentation and research literacy.

To evaluate the level of a chess player the community mainly relies on a number called the Elo rating. Given two players with Elo ratings Elo_A and Elo_B, the expected score (probability of A winning) is:

$$E_A = \frac{1}{1 + 10^{(\text{Elo}_B - \text{Elo}_A)/400}} \qquad (2)$$

This is the standard Elo formula, calibrated for chess [9]. While this metric captures relative strength between two players, its absolute interpretation depends heavily on methodology and context. Community conventions suggest that players understanding the basic rules rate between 600 and 900 Elo, casual players typically orbit around 1300 Elo, and from 2000 Elo onward, players enter the master and grandmaster territory [9]. At his peak in May 2014, Magnus Carlsen achieved a classical rating of 2882, the highest in history, and he also achieved 2909 in the newly introduced Freestyle Chess (Chess960) rating system [10].

2.3 Empirical Studies of Crowd Chess

The Kasparov vs. The World match (1999) is the earliest documented crowdsourced chess challenge. Over four months, approximately 50,000 amateur and intermediate players voted on moves against the then-World Champion. Despite reaching an advanced position, the crowd eventually conceded after 62 moves when facing maximum resistance from Kasparov [14]. While historically significant, this single-game format provides weak evidence: a single outcome cannot be reliably attributed to collective intelligence versus chance.

The Fouloscopie experiment rectified this limitation through large-scale repetition [20]. Moussaïd and colleagues organized approximately 25,000 human participants to play a total of 500 games against chess engine opponents. Key design features included:

- **Diverse player population**: Participants self-reported their Elo rating; the population was approximately normally distributed with a mean around 1165 Elo.
- **Majority-vote aggregation**: Players had several hours to evaluate positions and vote on the next move; the most-voted move was played automatically.
- **Opponent variety**: Chess engine opponents (Maia models) ranged from weak (1100 Elo) to strong (1900 Elo), allowing measurement of performance across the strength spectrum.
- **Social influence manipulation**: Comparing voting-only versus poll-visibility conditions revealed how information visibility affects collective decision quality.

Results showed that crowds achieved a win rate exceeding 60% across all tested opponents. With poll visibility, win rates improved to approximately 77%, while visibility-hidden conditions showed approximately 64% win rates. Importantly, the crowd played measurably better than the average individual member, confirming classic wisdom-of-crowds predictions.

2.4 Chess Engines

Modern chess engines span multiple paradigms, each with distinct decision-making processes. We focus on three particularly relevant approaches.

2.4.1 Stockfish. Stockfish, as of 2025, is among the world's strongest chess engines. It is an open-source tree-search algorithm combining classical techniques (alpha-beta pruning, endgame tables) with small neural networks for position evaluation [26]. Its strength is calibrated via Elo ratings: users can set `UCI_Elo` to values in an approximate range from 1320 to 3190. The engine deliberately weakens its play through stochastic evaluation perturbation (seeded random noise), ensuring exploration of different move trees at reduced strength levels.

2.4.2 Maia: Human-Mimetic Neural Networks. In contrast to optimal play, Maia learns to *predict human moves* from millions of Lichess games [18]. Rather than computing optimal moves, Maia generates a probability distribution over legal moves, parameterized by player skill level. Maia-2, introduced as a follow-up architecture, incorporates a skill-aware attention mechanism that dynamically integrates player skill level with board position encoding [28]. This mechanism achieves approximately 50–52% top-1 move accuracy across human players rated 1100 to 1900 Elo. In practical play, Maia replicates human decision-making nearly indistinguishably, making it an excellent tool for evaluating crowd performance against human-like opponents.

2.4.3 Mad Chess. MadChess is a classical alpha-beta search engine similar to Stockfish. Written in C#, its latest release (v3.3) reaches a playing strength above 2800 Elo in its strongest configuration and can easily perform over 2400 even cap at a fraction of a second per move [17]. What makes MadChess particularly attractive for experiments requiring a wide and realistically calibrated Elo spectrum, is its strength reduction algorithm. With parameters explicitly tuned for "realistic" weakened play, it mimic player with precise level going as low as 600 Elo. We leverage theses capabilities in our ensemble experiments (Section 4.1).

2.4.4 Large Language Models. Recent work demonstrates that Large Language Models (LLMs) trained on chess game transcriptions can develop emergent internal representations of board states [13]. These representations can be altered, thereby changing the move predicted by the LLM. This represents a fundamentally distinct paradigm: LLMs make decisions based on statistical pattern recognition from training data rather than explicit search or neural evaluation. Their move quality is unstable and depends on many factors (model architecture, PGN-based prompting, board representation, move history), which are often difficult to replicate or interpret.

Prelec's "single-question crowd wisdom" result [23] and related work suggest that aggregating judgments with access to meta-predictions or rationales can be especially powerful, which motivates using LLMs as both players and commentators in chess ensembles.

But recent work such as LLM CHESS introduces a dynamic benchmark that evaluates reasoning and instruction-following in LLMs via full chess games against configurable opponents, reporting that even state-of-the-art reasoning models often struggle to reliably win or complete games. [15] Carlini showed that `gpt-3.5-turbo-instruct` can achieve a playing strength of roughly 1788 Elo under specific conditions [4]. More recent transformer-based models have achieved grandmaster-level performance when trained directly on game transcriptions [24].

Although modern chat-oriented LLMs appear to lag behind specialized models, Dynomight [8] demonstrated that prompting strategies (such as requiring the model to regenerate the move history before outputting the next move) can dramatically improve their move quality. These results suggest that apparent "unusual" chess behavior in some LLMs may stem less from intrinsic chess competence and more from specific training setups and prompt configurations. For ensemble experiments, LLMs offer substantial cognitive diversity and potential for open deliberation, but also present engineering challenges that must be carefully controlled (API management, prompt optimization, legal move filtering).

2.5 Aggregating Engines

Carvalho et al. [5] studied majority voting among homogeneous groups of checkers engines. They found that group performance improves roughly logarithmically with group size, while outcome variance decreases. In chess, Spoerer et al. [25] investigated three-member simple majority voting among engines of varying strength. They analyzed how different configurations (e.g., equal-strength engines versus mixtures of stronger and weaker ones) affect ensemble performance and the variance of game outcomes. Their results indicate that even very small committees can benefit from majority aggregation, but that gains depend sensitively on the diversity and relative strength of the members, as well as on how ties and disagreements are resolved. According to this study, to observe a stronger wisdom-of-crowds effect, one should focus on low-Elo engines with small strength differences, which seems to contradict the diversity assumption.

Another team have explicitly optimized engines for *skill-compatibility* in collaborative chess settings, where near-optimal engines are adapted to cooperate effectively with much weaker partners rather than simply maximizing standalone performance [12].

3 METHODOLOGY AND FRAMEWORK

ChessLab is a modular, open-source framework for large-scale chess engine evaluation and crowd experimentation. In this section, we detail some of its inner workings and base modules.

3.1 UCI Engines

Chess engines communicate with an interface named the Universal Chess Interface (UCI), an open text-based protocol that standardizes how engines and user interfaces exchange commands and options, and is now supported by most modern chess programs [19].

We leverage this protocol to integrate any UCI-compliant engine, such as Stockfish. For engines or models that do not expose UCI (for example, Maia), we provide a custom wrapper interface. In addition, we implement several synthetic agents internally (e.g., random baselines and LLM-based agents). By exposing all engines through a common abstraction, we can treat them uniformly and combine them into a "crowd" engine governed by a configurable aggregation rule.

3.2 Aggregator Rules

We define a family of aggregation rules that map a set of engine proposals into a single move choice. Examples include:

- **Majority**: Each engine proposes a single move; the move with the highest vote count is played, with ties broken randomly.
- **Minority**: The least popular move among the proposed ones is selected, intended as an extreme contrarian baseline.
- **Randomized**: One of the proposed moves is sampled at random, with or without weighting by engine Elo.
- **Top-Elo dictator**: The move proposed by the highest-rated engine in the ensemble is always selected.
- **Bottom-Elo dictator**: Symmetric to the top-Elo dictator, using the lowest-rated engine; serves as a control for "anti-expert" aggregation.
- **Median-Elo dictator**: The move from the median-rated engine is chosen, approximating a representative member of the group.
- **Rotating dictator**: Engines take turns acting as dictator across moves or games, ensuring equal participation over time.
- **Elo-weighted**: Each engine casts a vote weighted by a function of its Elo (e.g., proportional to Elo, or logistic in Elo); the move with the highest total weight is selected.

These rules are not intended to lead to optimal results, but rather to be compared against each other and to test their effectiveness in artificial crowds relative to human crowds. Their formalism come from the voting and preference aggregation studies of computational social choice sciences [3].

3.3 Game Execution Engine

The core of ChessLab is a Python-based game runner built on top of the `python-chess` library, which manages internal move generation and game-state transitions. Games are executed asynchronously using `asyncio`, allowing highly parallelized evaluation across many engine-versus-engine matchups. The PostgreSQL schema is organized around the following core entities:

- **Players**: Engine type (e.g., Stockfish, Maia, LLM, random), name, nominal Elo rating, creation timestamp, and configuration parameters such as depth limits or temperature settings.
- **Games**: References to white and black player IDs, the numeric result (1–0, 0–1, 1/2–1/2), full PGN, and metadata about the game.
- **Moves**: For each game, a sequence of moves with SAN and UCI representations and the FEN preceding each move.
- **Evaluations**: For a given move, the required engines (player engine or multiple engines in case of a crowd) produce an evaluation (in centipawns or similar score).

3.4 Statistical Analysis Module

ChessLab evaluates engine and ensemble performance through statistical analysis at multiple levels of granularity: individual games, pairwise matches, and multi-opponent campaigns. The framework implements hypothesis testing procedures to distinguish genuine performance differences from random variation, and it provides Elo rating estimation with uncertainty quantification.

3.4.1 *Per-Game Metrics.*

For each completed game, ChessLab computes both outcome-based and quality-based metrics. The outcome is encoded as a score $s \in \{1, 0.5, 0\}$ representing win, draw, or loss respectively. Beyond win/loss statistics, we evaluate move quality through centipawn loss analysis, where each move's quality is assessed by comparing the position evaluation before and after the move using Stockfish at depth 10. For a player who made N moves in a game, the average centipawn loss is:

$$\mathrm{ACPL} = \frac{1}{N}\sum_{i=1}^{N} |\mathrm{eval}_{\mathrm{before}_i} - \mathrm{eval}_{\mathrm{after}_i}| \tag{3}$$

where evaluations are reported from the moving player's perspective. This metric provides an independent assessment of play quality that complements outcome-based statistics, particularly useful for fast evaluation on specific positions.

3.4.2 *Elo Rating Estimation.*

ChessLab estimates Elo ratings using the standard logistic model.

Single-Opponent Estimation. When a player achieves mean score $\bar{s}$ over n games against a single opponent with known rating R_{opp}, we invert equation (2) to obtain:

$$\hat{R} = R_{\mathrm{opp}} + 400\log_{10}\left(\frac{\bar{s}}{1-\bar{s}}\right) \tag{4}$$

To avoid numerical issues at the boundaries, we clip observed scores to the range $[\epsilon, 1-\epsilon]$ where $\epsilon = 10^{-9}$, and constrain the final estimate to $[250, 3000]$ Elo.

Multiple-Opponent Estimation. When evaluating against m opponents with known ratings $R_1, \ldots, R_m$ and corresponding observed scores $\bar{s}_1, \ldots, \bar{s}_m$ from $n_1, \ldots, n_m$ games, we estimate the player's Elo by minimizing the weighted squared error between observed and expected scores:

$$\hat{R} = \arg\min_{R\in[250,3000]} \sum_{i=1}^{m} w_i(\bar{s}_i - E(R, R_i))^2 \tag{5}$$

where the weights are normalized as $w_i = n_i / \sum_{j=1}^{m} n_j$. This objective function is optimized using bounded scalar minimization with a tolerance of 0.1 Elo points. The resulting estimate $\hat{R}$ represents the single rating that best explains the ensemble of results across all opponents, accounting for both the strength of each opponent and the number of games played.

3.4.3 *Statistical Testing.*

ChessLab performs hypothesis testing at two levels: single matches against one opponent and ensemble evaluation across multiple opponents. Both use the same underlying statistical framework but differ in how mean scores and standard errors are computed.

Single-Match Analysis. A match consists of n games between two fixed players. Given individual game scores $s_1, s_2, \ldots, s_n$, the mean score is:

$$\bar{s} = \frac{1}{n}\sum_{i=1}^{n} s_i \tag{6}$$

For an expected score p (derived from Elo ratings via equation (2)), the standard error is:

$$\mathrm{SE}(p) = \sqrt{\frac{p(1-p)}{n}} \tag{7}$$

Multi-Opponent Ensemble Analysis. When evaluating against m opponents with Elo ratings $R_1, R_2, \ldots, R_m$, where match i consists of n_i games with observed mean score $\bar{s}_i$ and expected score p_i, we compute weighted ensemble statistics. The ensemble observed mean score is:

$$\bar{s}_{\text{ensemble}} = \frac{\sum_{i=1}^{m} n_i \bar{s}_i}{N}, \quad \text{where } N = \sum_{i=1}^{m} n_i \tag{8}$$

The ensemble expected mean is computed analogously:

$$\bar{p}_{\text{ensemble}} = \frac{\sum_{i=1}^{m} n_i p_i}{N} \tag{9}$$

To account for heterogeneous expected scores across matches, the standard error of the pooled proportion is:

$$\mathrm{SE}_{\text{ensemble}} = \frac{\sqrt{\sum_{i=1}^{m} n_i p_i (1-p_i)}}{N} \tag{10}$$

This formulation correctly handles cases where different matches have different expected outcomes, which is essential when testing an engine against a ladder of opponents at varying strength levels.

Hypothesis Testing. For both single-match and ensemble analyses, the test statistic is:

$$z = \frac{|\bar{s} - p|}{\mathrm{SE}(p)}, \tag{11}$$

where $\bar{s}$ and p represent either the single-match or ensemble values as appropriate. We automatically select between z-test and t-test based on sample size and expected frequency. For single matches, when $np \geq 5$ and $n(1-p) \geq 5$, we use the normal approximation (z-test); otherwise we use the t-distribution with $n-1$ degrees of freedom. For ensemble analysis, we apply a conservative check: if $\min_i(n_i p_i) \geq 5$ and $\min_i(n_i(1-p_i)) \geq 5$, we use the z-test; otherwise we use a t-distribution with $N-m$ degrees of freedom. The two-tailed p-value quantifies the probability of observing results as extreme as $\bar{s}$ under the null hypothesis that the player's true strength matches expectation. We use this testing framework throughout ChessLab to assign estimated Elo ratings across heterogeneous agent types on a unified scale.

4 PROOF-OF-CONCEPT EXPERIMENTS

In this section we present preliminary experiments designed to validate ChessLab's Elo calibration procedures and demonstrate basic ensemble behaviors. Since there is no absolute standard for engine Elo ratings, we first test whether engines are internally coherent with respect to their own Elo settings. We then establish a practical reference standard for cross-engine comparisons and demonstrate collective intelligence phenomena. These results serve as methodological validation and motivate the more extensive roadmap outlined in Section 5.

The figures are auto-generated by the statistical analysis module. Columns represent the measured scores obtained in practice across different strength levels. Green represents the victory ratio, red the loss ratio, and white the draw ratio. Curves, with a specified Elo, represent the expected mean score (abscissa) for a player of that Elo against an opponent of a given Elo (ordinate).

4.1 Elo Coherence Experiments

To establish a reliable foundation for comparing engines across different architectures, we evaluate the internal Elo coherence of three diverse engines:

Stockfish. Stockfish demonstrates strong internal coherence when configured at depth 10. Across the tested strength levels, the estimated Elo values align closely with the nominal calibration, as shown in Figure 3 in Appendix 6.

MadChess. Similarly in Figure 1, MadChess exhibits robust Elo coherence across its exceptionally wide rating range (600–2600 Elo) when configured at 0.3 seconds per move. This broad coverage makes MadChess particularly valuable for testing weak agents and diverse ensembles.

Maia2. In contrast, Maia2 displays less consistent Elo coherence across its supported range (1100–1900 Elo) when run without explicit time constraints, as shown in Figure 4 in Appendix 6. This finding may tentatively indicate a potential issue with the implementation or calibration suggested by the original authors, though we note that the Maia team is actively developing Maia3 with wider Elo coverage. Further investigation is needed to determine whether these deviations stem from our experimental setup or the engine's intrinsic characteristics.

Cross-Engine Validation. To verify that different engines' Elo scales are mutually consistent, we conducted head-to-head matches between Stockfish and MadChess. The results, provided in Figure 5 in Appendix 6, show some agreement between their rating systems. Stockfish seams to be underperforming by up to 200 Elo in lower lever but getting closer by to the expectation in higher level. Given this alignment, combined with MadChess's speed, superior Elo range and demonstrated coherence, we adopt MadChess as our practical reference standard for subsequent experiments. This choice enables testing of very weak agents while maintaining reliable strength calibration across the full rating spectrum.

4.2 Crowd Ensembles

Having established MadChess as our reference standard, we construct an ensemble of 10 MadChess instances with Gaussian-distributed

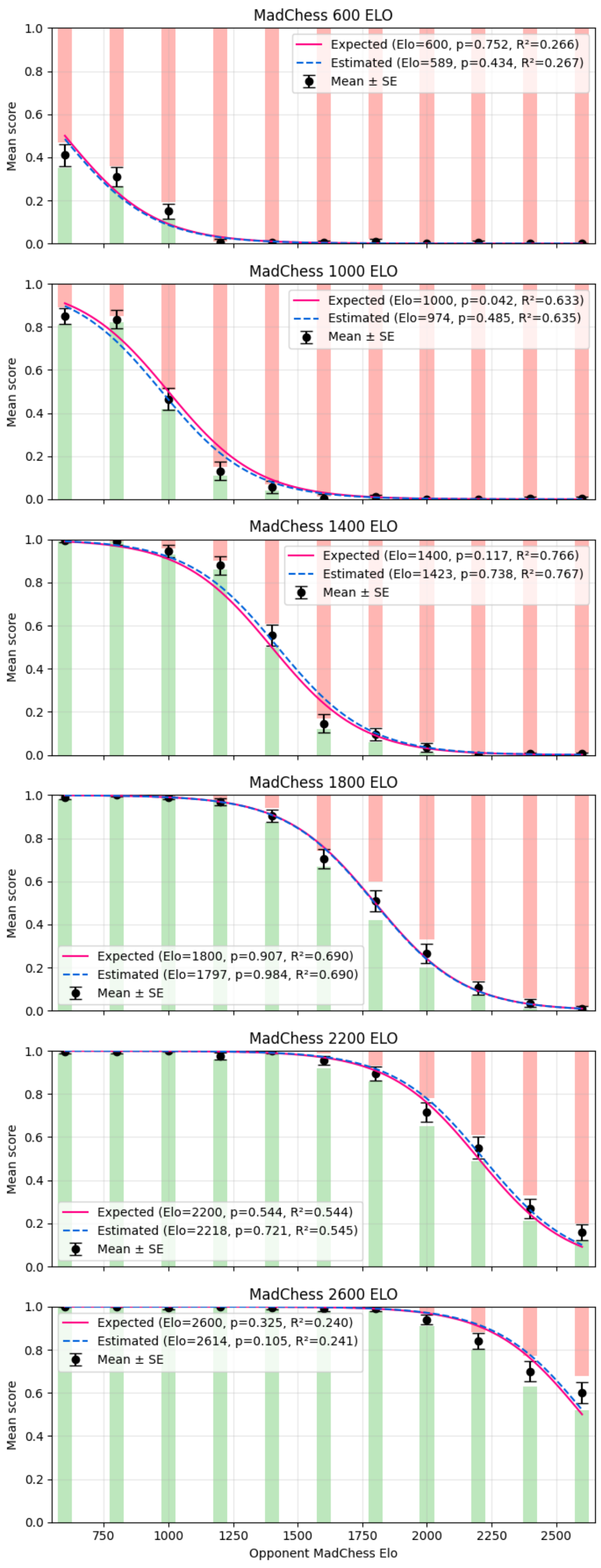

Figure 1: Elo estimation for a range of MadChess engines (0.3 second per move)

strength (mean 1100 Elo, standard deviation 200 Elo) and evaluate various aggregation rules against a ladder of seven MadChess opponents spanning 800–2000 Elo. To permit rapid iteration, we limit this proof-of-concept to 20 games per opponent.

Results are illustrated in Figure 2. Preliminary observations suggest that majority voting with its estimated 1202 Elo can significantly outperform the average individual member (1048 Elo for this experiment). Other aggregation methods (randomized ballot, Elo-weighted variants, rotating dictator) yield similar though less performant profiles.

These results suggest that even simple majority voting among a small but diverse group of weaker engines can achieve collective intelligence gains in this domain. The question is now how much gain can be leverage.

4.3 Local LLM

Finally, we test three local LLMs with naive prompting:

- Meta Llama-3.2-1B-Instruct
- Qwen Qwen2.5-1.5B-Instruct
- Google Gemma-3-1B-IT

Board states were encoded in plain algebraic notation (PAN) with the prompt "Given the following chess position, what is the best legal move? ". In this setting, the LLM agents failed to win a single game across all opponents, and average loss per move exceeded 1800 centipawns. These results highlight both the potential and the challenges of incorporating LLM-based agents into ChessLab: substantial prompt engineering (e.g., move history reconstruction [8]), legality filtering, and domain-specific fine-tuning are required before LLMs can serve as a measurable and reliable member of the crowd.

5 ROADMAP

The current work establishes ChessLab's core infrastructure, validates its Elo calibration methodology through coherence experiments, and demonstrates basic ensemble feasibility. However, many key questions about artificial collective intelligence in chess remain open. We outline here a roadmap of planned experiments and framework extensions.

Planned experiments include:

- Exploring the variation on engines, size and sampling distribution on the crowd ensembles experiments.
- Calibrating distilled Stockfish and Maia variants to obtain lower-Elo opponents. Distilled engines use stochastic weakening mechanisms (e.g., random-move probabilities) to benchmark very weak chess engines [21]. We also hope to propose a model of byzantine fault tolerance in chess playing crowd.
- Reproducing the Moussaïd experiment as closely as possible, using estimates of participant strength, and comparing human and artificial crowds under analogous conditions [20].
- Systematically exploring LLM prompting strategies (e.g., reconstructing move history, chain-of-thought reasoning) to improve LLM move quality and stability [8].
- For sufficiently capable LLMs, attempting to assign effective Elo ratings and integrating them into mixed-architecture ensembles.

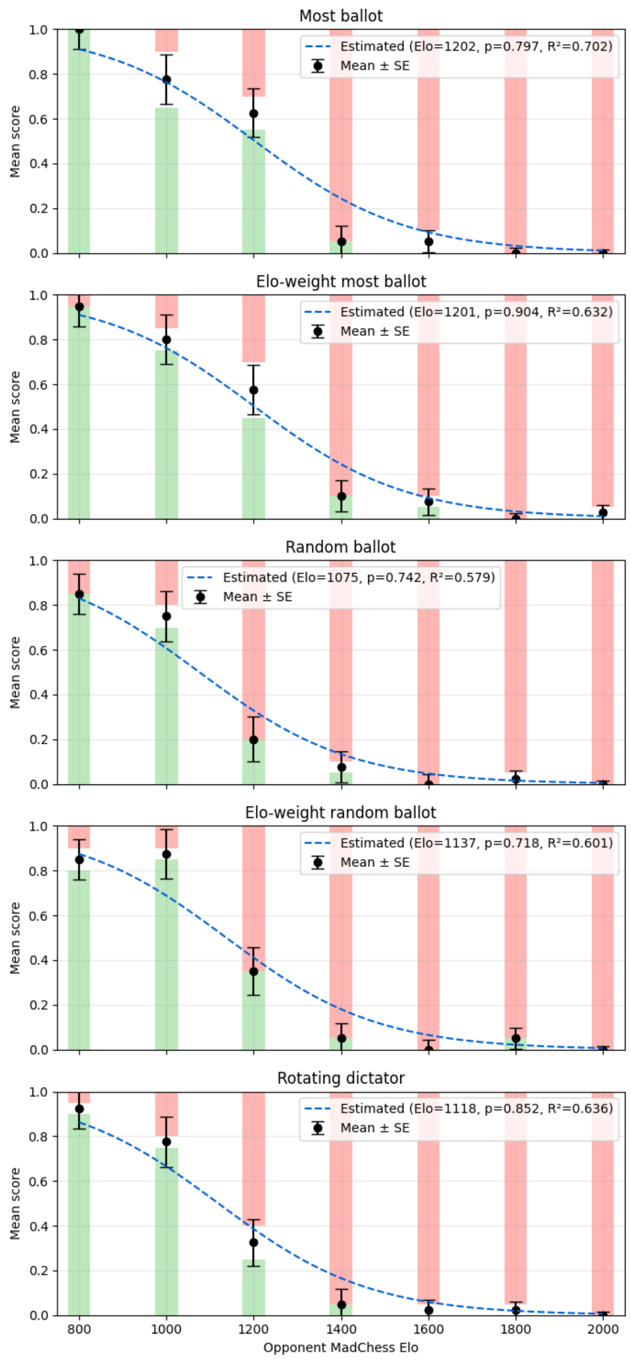

Figure 2: Performance of a MadChess crowd with variations on aggregation methods

- Having LLMs output their top-k candidate moves with rationales, enabling more sophisticated aggregation rules (e.g., Condorcet, Borda, Coombs, approval voting).
- Studying information-sharing regimes, such as letting LLMs observe current poll results before voting, to simulate social influence dynamics.
- Investigating argumentation pipelines in which multiple LLMs propose moves and arguments, and a separate "arbiter" LLM selects the final move.

Planned framework extensions include:

- Adding tactical puzzle modes to estimate Elo via centipawn loss on curated positions, accelerating preliminary testing on LLM-based engines.
- Integrating external API-based LLMs, with rate-limit-aware scheduling in the request queue.
- Providing containerized deployments (e.g., Docker images) for reproducible local and cloud experiments.
- Developing a web interface to share experiments with a broader audience and facilitate crowdsourced research contributions.

6 CONCLUSION

This paper introduces ChessLab, a framework for studying collective intelligence in chess by replacing human crowds with ensembles of artificial agents. The system enables hundreds of reproducible, low-cost experiments that test hypotheses from wisdom-of-crowds theory in a complex, sequential decision-making domain. By instrumenting games with detailed move-level evaluations and robust Elo estimation, ChessLab makes it possible to disentangle ensemble benefits arising from diversity, bias correction, and variance reduction, and to compare artificial and human crowds on a common quantitative scale. The roadmap from baseline replication of human experiments to optimization of voting mechanisms and information structures is designed to progressively bridge empirical chess results and broader theories of collective intelligence.

We release ChessLab as open-source software and invite the community to extend, replicate, and build upon this work: https://github.com/Uspectacle/chesslab Future publications will report on the larger set of experiments outlined here and on the resulting refinements to our understanding of artificial and human crowd wisdom in strategic games.

DISCLAIMER

The assistance of Large Language Models (LLMs) was used in a limited and controlled manner during the writing process of this paper. All scientific claims, experimental designs, theoretical contributions, and the final version of this paper remain the responsibility of the authors.

REFERENCES

[1] Abdullah Almaatouq, Alejandro Noriega-Campero, Abdulrahman Alotaibi, P. M. Krafft, Mehdi Moussaïd, and Alex Pentland. 2020. Adaptive Social Networks Promote the Wisdom of Crowds. *Proceedings of the National Academy of Sciences* 117, 21 (2020), 11379–11386.

[2] Pavel Atanasov, Phillip Rescober, Eric Stone, Samuel A. Swift, Emile Servan-Schreiber, Philip Tetlock, Lyle Ungar, and Barbara Mellers. 2017. Distilling the Wisdom of Crowds: Prediction Markets vs. Prediction Polls. *Management Science* 63, 3 (2017), 691–706.

[3] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. 2016. Introduction to Computational Social Choice. In *Handbook of Computational Social Choice*. Cambridge University Press, 1–29. https://doi.org/10.1017/CBO9781107446984.002

[4] Nicholas Carlini. 2023. Playing Chess with Large Language Models. https://nicholas.carlini.com/writing/2023/chess-llm.html. Accessed February 2026.

[5] Danilo S. Carvalho, Minh Le Nguyen, and Hiroyuki Iida. 2017. An Analysis of Majority Voting in Homogeneous Groups for Checkers: Understanding Group Performance Through Unbalance. In *Advances in Computer Games*. Springer, 213–223.

[6] Chess.com. 2025. GothamChess vs. The World. https://www.chess.com/news/view/gothamchess-vs-world-200k-players. Accessed February 2026.

[7] Chess.com. 2025. Magnus Carlsen Held to Draw by 143,000 Players. https://www.chess.com/news/view/the-world-forces-draw-in-historic-game-vs-magnus-carlsen. Accessed February 2026.

[8] Dynomight. 2024. OK, I can partly explain the LLM chess weirdness now. https://dynomight.net/more-chess/ Accessed February 2026.

[9] Arpad E. Elo. 1978. *The Rating of Chessplayers, Past and Present.* Arco Publishers.

[10] FIDE. 2025. Magnus Carlsen Chess Ratings. https://www.2700chess.com/players/carlsen_magnus. Live rating; Accessed February 2026.

[11] Francis Galton. 1907. Vox Populi. *Nature* 75, 1949 (1907), 450–451.

[12] Karim Hamade, Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. 2024. Designing Skill-Compatible AI: Methodologies and Frameworks in Chess. *arXiv preprint arXiv:2405.05066* (2024). https://doi.org/10.48550/arXiv.2405.05066

[13] Anssi Karvonen. 2024. Emergent World Models in Chess-Playing Language Models. *arXiv preprint* (2024). arXiv:2403.15498 [cs.AI]

[14] Garry Kasparov. 1999. Kasparov versus the World. *The New York Times Magazine* (1999).

[15] Sai Kolasani, Maxim Saplin, Nicholas Crispino, Kyle Montgomery, Jared Quincy Davis, Matei Zaharia, Chi Wang, and Chenguang Wang. 2025. LLM CHESS: Benchmarking Reasoning and Instruction-Following in LLMs through Chess. *arXiv preprint arXiv:2512.01992* (2025). https://doi.org/10.48550/arXiv.2512.01992

[16] Asher Koriat. 2011. Subjective Confidence in Perceptual Judgments: A Test of the Self-Consistency Model. *Journal of Experimental Psychology: General* 140 (2011), 117–139.

[17] Erik Madsen. [n.d.]. MadChess Chess Engine. https://www.madchess.net/. Accessed February 2026.

[18] Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. 2020. Aligning Superhuman AI with Human Behavior: Chess as a Model System. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2020), 1677–1687.

[19] Stefan Meyer-Kahlen and Rudolf Huber. 2000. UCI: Universal Chess Interface. https://www.shredderchess.com/chess-features/uci-universal-chess-interface.html. Accessed February 2026.

[20] Mehdi Moussaïd and Simon Garnier. 2022. Mechanisms of Collective Intelligence in Chess. Conference presentation at the ACM Collective Intelligence Conference 2022. Extended findings discussed in *A-t-on besoin d'un chef? Petit traité d'intelligence collective* (2025).

[21] Tom Murphy VII. 2019. Elo World, a Framework for Benchmarking Weak Chess Engines. In *Proceedings of SIGBOVIK 2019.* 1–13. https://tom7.org/chess/weak.pdf

[22] Scott E. Page. 2007. *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies.* Princeton University Press.

[23] Drazen Prelec, H. Sebastian Seung, and John McCoy. 2017. A Solution to the Single-Question Crowd Wisdom Problem. *Nature* 541, 7638 (2017), 532–535.

[24] Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Li Kevin Wenliang, Elliot Catt, John Reid, Cannada A. Lewis, Joel Veness, and Tim Genewein. 2024. Amortized Planning with Large-Scale Transformers: A Case Study on Chess. arXiv:2402.04494 [cs.AI] arXiv preprint.

[25] Kristian Toby Spoerer, Toshihisa Okaneya, Kokolo Ikeda, and Hiroyuki Iida. 1999. Further Investigations of 3-Member Simple Majority Voting for Chess. In *Computers and Games, CG 1998 (Lecture Notes in Computer Science, Vol. 1558).* Springer.

[26] Stockfish Development Team. 2024. Stockfish Documentation. https://stockfishchess.org/docs. UCI protocol and engine specifications; Accessed February 2026.

[27] James Surowiecki. 2004. *The Wisdom of Crowds: Why the Many Are Smarter than the Few and How Collective Wisdom Shapes Business, Economies, Societies, and Nations.* Vintage.

[28] Zhenwei Tang, Difan Jiao, Reid McIlroy-Young, Jon Kleinberg, Siddhartha Sen, and Ashton Anderson. 2024. Maia-2: A Unified Model for Human-AI Alignment in Chess. arXiv:2409.20553 [cs.AI] arXiv preprint.

A SUPPLEMENTARY CALIBRATION PLOTS

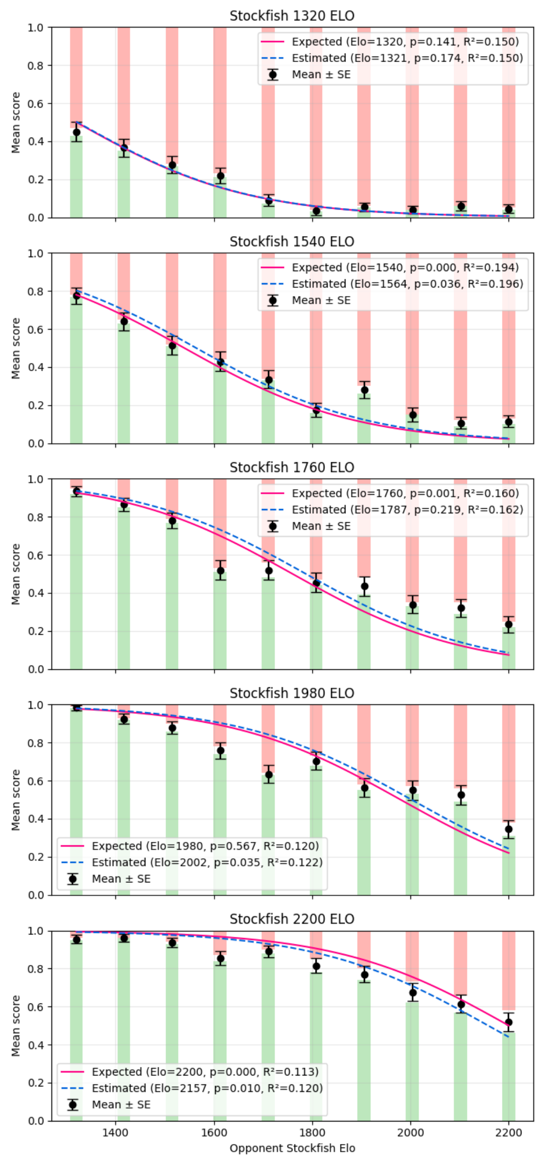

Figure 3: Elo estimation for a range of Stockfish engines (depth 10)

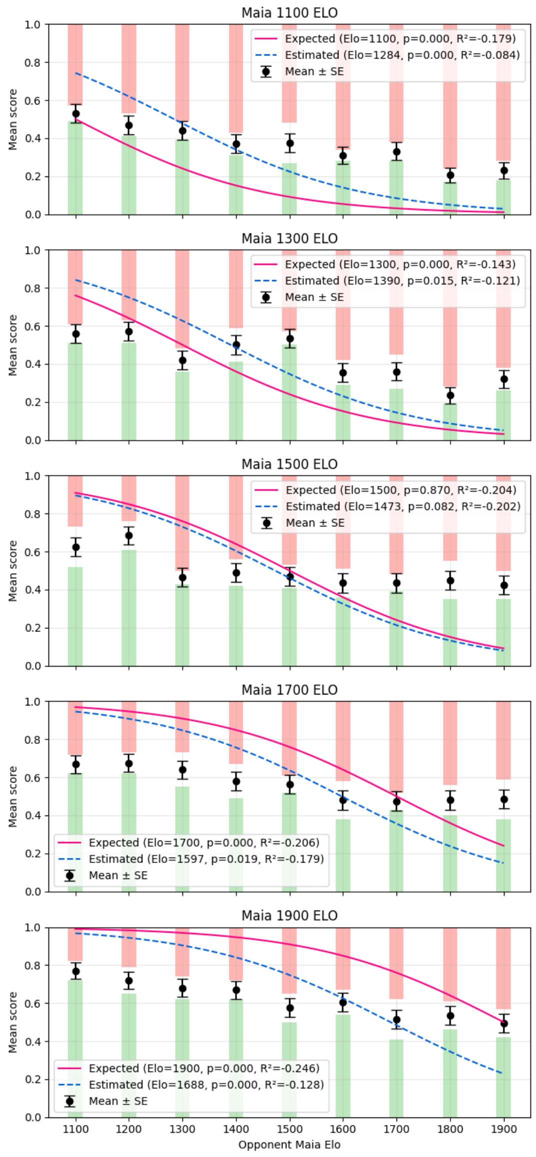

Figure 4: Elo estimation for a range of Maia engines

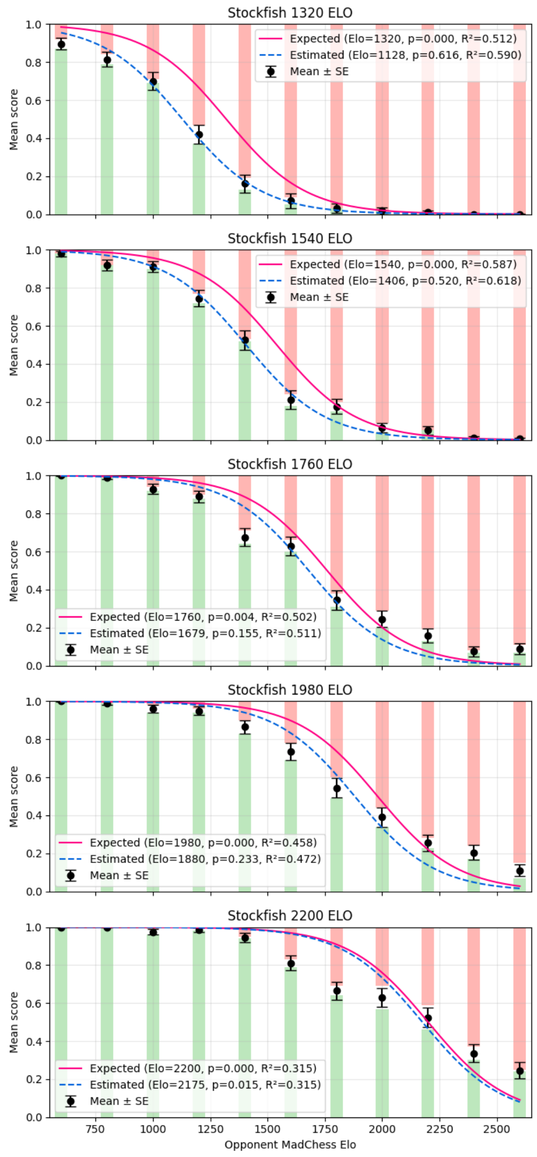

Figure 5: Elo estimation for cross-validation between Stockfish and MadChess