ChatGPT

Phase 8 Plan: Scorpion Chess Engine

Overview

Phase 8 focuses on elevating Scorpion's strength and consistency by fully integrating a neural network (NNUE) as the primary evaluation, while refining the engine's strategic playstyle. This phase emphasizes defensive solidity – minimizing tactical blunders and material losses – and deeper long-term planning in line with classical chess principles. We will leverage high-quality game data (PGNs from Scorpion vs. Stockfish matches and GM-level games) to guide training and adjustments. All improvements will be designed to fit Scorpion's existing roadmap and architecture (e.g. parallel search, MCTS, endgame bitbases), ensuring continuity with previous phases and the engine's core goals [1] [2]. The plan is structured into clear milestones with implementation hints and measurable benchmarks.

Milestone 1: Full NNUE Integration as Main Evaluation

Objective: Replace the traditional evaluation function with a fully-integrated NNUE network, making the neural net score the engine's primary way to judge positions. This yields a stronger, more nuanced evaluation and aligns with modern engine trends [3] [4]. (Scorpio's developer reported a **~400 Elo gain** after adding NNUE [5], underscoring the importance of this integration.)

Implementation Steps:
- **Embed NNUE Network:** Incorporate an efficient, fully-connected NNUE model into Scorpion's evaluation pipeline. Use the established NNUE input format (e.g. piece-square feature indices, possibly differentiated by king safety or side-to-move) to encode board state [6]. This preserves classic positional knowledge (material, pawn structure, king zone, etc.) in the input features so the network can learn these patterns.
- **Efficient Inference:** Implement the NNUE with **incremental update** logic and SIMD optimizations to maintain high node throughput [7] [8]. When a move is made or undone, update the neural net's hidden layer accumulator for only the changed pieces instead of recomputing from scratch. This ensures the engine still searches millions of positions per second with minimal slowdown.
- **Training & Calibration:** Initialize the NNUE with a strong pretrained network (e.g. one trained on millions of positions) and fine-tune it on Scorpion's own game data. Specifically, train on a large set of positions evaluated by a high-depth engine (Stockfish) to "distill" evaluation knowledge [9]. This supervised training (using engine evaluations as ground truth) will teach the network to predict accurate centipawn scores for given positions. Supplement this with self-play or reinforcement learning games (Scorpion NNUE vs. earlier versions) to further calibrate the network on Scorpion's playstyle.
- **Integration into Search:** Make NNUE the default evaluation in both alpha-beta and any Monte Carlo Tree Search components. All search threads and cluster nodes should query the NNUE for static evals at leaf nodes. Ensure the legacy evaluation terms (material tables, PSTs, etc.) are either incorporated into the network or disabled to avoid conflict. The transition should be smooth since Scorpion's architecture already supports neural eval calls via the `egbbdll` module [5].
- **Validation:** Verify that the NNUE-integrated engine matches or exceeds the strength of the previous phase on a diverse test suite. Check that no major evaluation term was lost in the transition – e.g., test specific scenarios for which classical eval had known heuristics (pawn structure, outpost knights, etc.) to ensure the

NNUE output aligns with correct chess intuition. If any critical knowledge is missing, consider injecting it (via training data or minor hybrid eval logic for edge cases, such as using tablebases in late endgames).

Milestone 2: Strengthening Defensive Play (Blunder Reduction & Material Preservation)

Objective: Adjust Scorpion's search and evaluation behavior to be more risk-averse and airtight defensively. The goal is to **minimize tactical blunders** (moves that drop the evaluation by a large amount due to overlooking tactics) and to preserve material unless a sacrifice is clearly justified. A more defensive engine will hold draws against stronger opponents more often and only take calculated risks. This directly addresses Phase 8's priority of reducing costly mistakes.

Implementation Ideas:
- **Search Safeguards:** Introduce search modifications that specifically target blunder reduction. For example, implement **extended search in critical positions**: if the side to move has only one or two moves that avoid immediate catastrophe (e.g. only one move saves a piece or stops mate), apply a "one-reply" extension to search deeper in that line [10]. Likewise, always extend when the king is in check to avoid missing forced mates or escapes. These extensions ensure the engine sees at least a few extra plies in dangerous situations, catching tactics that could be overlooked.
- **Tactical Blunder Checks:** Add a verification mechanism for moves that appear promising but could hide a blunder. For instance, if a move causes a significant drop in the NNUE evaluation during search (say > 100 centipawns), trigger a re-search or quiescence search at higher depth for that move to double-check tactical sequences. This acts as a safety net so the engine doesn't choose a superficially good move that actually loses material on the next turn.
- **Late-Move Reduction Tuning:** Calibrate pruning heuristics like Late Move Reductions (LMR) to be more conservative in sharp positions. In practice, **do not reduce moves at shallow depths** (or in volatile tactical positions) as aggressively as before – for example, some engines disabled LMR for the first 3 plies to avoid pruning critical responses [11]. By ensuring full-depth evaluation of more moves in the early stages of search, Scorpion can catch tactics that a heavy reduction might skip. This tweak was shown to significantly improve playing strength in similar engines by avoiding shallow tactical oversights [11].
- **Material Safety Bias:** Adjust evaluation (or network training) to slightly overweight material advantage and piece safety when the engine is ahead, reinforcing a "don't drop what you've gained" mentality. For the NNUE, this can be achieved by training on positions emphasizing that losing a large amount of material leads to very poor evaluations. Within search, consider a **"safe move" heuristic**: when multiple moves have close evaluations, prefer the one that is materially safer (e.g. avoids sacrificing a piece) unless a clear tactical win is found. This might involve a secondary sort key for moves – giving a bonus to moves that do not entail immediate material risk.
- **Contempt and Draw Bias:** If not already present, introduce a mild **contempt factor** tuned for defense. Contempt > 0 (favoring draws as black, for example) can make the engine choose solid lines instead of speculative attempts to win at all costs. This will increase the draw ratio against stronger engines. The contempt should be set carefully: enough to prefer safe continuations when the position is equal or slightly worse, but not so high that the engine refuses to capitalize on a winning chance.

Expected Outcome: These changes will make Scorpion much less blunder-prone. Top engines at super-GM strength *"reduce tactical blunders to near zero, and convert endings with machine precision"* [12], leading to very few decisive mistakes. Phase 8 pushes Scorpion in that direction – it should rarely leave pieces en prise or

fall for simple tactics. The engine may accept more draws as a trade-off for not losing, which is desirable for a defensive style. We will measure progress by tracking a drop in blunder rate (see **Benchmarking** section).

Milestone 3: Deeper Long-Term Positional Planning

Objective: Improve Scorpion's understanding of long-horizon positional factors and its ability to formulate plans that extend beyond immediate tactics. This involves ensuring the evaluation (now NNUE) properly values enduring advantages such as superior pawn structure, king safety, control of key squares, and potential outposts – all of which might only pay off after many moves. By strengthening positional play, Scorpion will make better strategic decisions (e.g. when to trade pieces, how to structure pawns, whether to accept weaknesses) that lead to favorable endgames or fortress defenses. This milestone aligns with Scorpion's existing architecture, which already recognized many of these features in its classical eval [13] [14], and now translates that knowledge into the neural era.

Implementation Ideas:
- **NNUE Feature Enhancement:** Ensure the NNUE network's input and training data emphasize positional features. For example, include inputs encoding **pawn structure patterns** (isolated pawn, passed pawn, doubled pawns) and **king safety** (pawn shield, open files near king) so the network can learn their impact. Many of these were part of the old eval (e.g. pawn structure hash, king safety terms, outpost bonuses) [15]. The NNUE's piece-square features inherently cover some of this (a piece on a strong square will contribute higher value if the net learned that pattern). We can also incorporate additional context in the input, such as piece locations relative to the enemy king (to encode attacks) [6]. Training positions should include a high proportion of slow, strategic games and endgames, so the net learns to value things like space advantage, bishop pair, or knight outposts even when no immediate tactics are present.
- **Positional Training and Tuning:** Use a curated set of GM-level games and engine games that exemplify long-term plans to fine-tune the evaluation. For instance, take positions from master games where one side has a structural advantage (better pawn structure, strong knight vs bad bishop, etc.) and ensure the NNUE evaluates those correctly (favorable to the side with the long-term edge). If the raw network doesn't adequately appreciate such factors, consider **blending a bit of handcrafted evaluation** as a temporary aid: e.g. adding a small bonus in eval for features like "protected passed pawn" or "king on an open file". These bonuses can guide the network until training fully absorbs them. The eventual goal is the NNUE alone captures these, but interim hybrid evals are acceptable if needed to avoid regressions in positional understanding.
- **Search Strategy for Planning:** Augment the search to support plan formulation. One idea is implementing a **multi-PV search** during analysis or testing phases to have the engine consider alternative plans. For example, generating two or three principal variations can help identify different strategic ideas (like a kingside attack vs. a queenside expansion) and evaluate which is stronger long-term. While the engine normally picks the single best move, having multi-PV as a tool for analysis can help in debugging whether the engine recognizes various plans. Additionally, incorporate **search depth strategy tuning**: in quiet positional scenarios (no tactical melee), allow the engine to allocate more time/depth, since these positions might require deeper understanding (this could be through a dynamic depth extension when the evaluation is close to 0 and no tactics are found, to explore subtle maneuvers).
- **Positional Play Testing:** Create a test suite of positional problems to evaluate improvement. These could include classic scenarios: e.g. *good knight vs bad bishop endgame*, *fortress positions*, *long-term pawn sacrifice for compensation*, *closed positions requiring pawn breaks*. Monitor how the Phase 8 engine handles these compared to Phase 7. We expect to see more patience (not rushing pawn moves, avoiding weakening its

structure) and better assessment of fortresses or blockades (the eval stays balanced if a fortress truly cannot be broken, rather than falsely trending upwards and overestimating a position).

By the end of this milestone, Scorpion's play should exhibit more "grandmaster-like" planning: it will pursue small advantages and solidify positions that favor it, instead of solely relying on tactics. Strong squares and pawn majorities will be nurtured into endgame advantages. This improved strategic IQ will also complement the defensive focus – the engine will be less likely to drift into bad long-term structures or fall for slow plans by the opponent because it sees them coming.

Milestone 4: Leverage Stockfish/GM Game Data for Strategic Tuning

Objective: Use insights from games against **Stockfish and Grandmasters** to guide further refinements in Scorpion's play. By analyzing PGN data from these high-level games, we can identify Scorpion's recurring weaknesses and adjust accordingly. This data-driven approach will ensure that Phase 8's changes are grounded in real-game scenarios and directly address issues observed in practice. In essence, we are using stronger opponents as a training partner/teacher – learning from losses or draws to improve Scorpion's decision-making. This aligns with Scorpion's roadmap ethos of incorporating learning and modern AI techniques into the engine's development.

Steps and Ideas:
- **Game Analysis Pipeline:** Develop a pipeline to automatically analyze Scorpion's games against Stockfish (or other top engines) and high-level human games. Use a strong engine (Stockfish at very high depth) to annotate each move with an evaluation and identify **significant mistakes**. For Scorpion's games, log any move where the evaluation swings heavily (e.g. a drop of >0.5 pawns) and classify it as a potential blunder or misjudgment. Summarize these instances across many games to see patterns – for example, perhaps Scorpion frequently mishandles opposite-colored bishop endings, or often overestimates an attack that Stockfish defends, etc.
- **Strategic Pattern Mining:** From GM games, extract key strategic themes (with the help of engine analysis). For instance, identify games where one side slowly outplayed the other without obvious blunders – these showcase strategic play. Compare Scorpion's evaluation of those positions to Stockfish's: does Scorpion agree that, say, a backward pawn is a long-term weakness, or that a knight outpost on d5 is worth a pawn? If not, this indicates a gap in the evaluation. Use these findings to adjust the NNUE training set or evaluation terms. For example, if Stockfish consistently values a certain pawn structure more than Scorpion, incorporate those positions (with Stockfish's eval as the target) into NNUE training so that Scorpion learns a similar appreciation [9].
- **Targeted NNUE Retraining:** Assemble a training dataset from the above analysis. This can include: positions where Scorpion blundered (with correct evaluations from Stockfish as labels), positions from GM games with Stockfish evals, and critical endgame positions (possibly labeled by tablebase outcomes for perfect play). Perform **supervised learning updates** on the NNUE network with this data to fine-tune its weights. The idea is to teach the net: *"in these types of positions, the correct evaluation is X"*. For example, if Scorpion tended to undervalue king safety leading to getting checkmated, provide many examples where leaving the king in the center was punished, so the net will assign a bigger penalty to unsafe king positions in the future. This approach uses expert data to **guide the neural net's strategic bias** in a direction proven successful by stronger play.
- **Adaptive Search Tuning:** In addition to training the eval, adjust search parameters based on game data.

If analysis shows Scorpion often lost won positions on time or depth issues, consider increasing time allocation or using adaptive move pruning in complex positions (time management). If many losses came from specific opening lines (particularly against Stockfish's sharp play), perhaps integrate an opening book or more opening knowledge to reach solid middlegames. Essentially, close the practical loops that raw NNUE training might not cover: time management, opening selection, and other non-eval factors gleaned from match experience.
- **Testing Against Reference Opponents:** After implementing changes, run a new gauntlet of matches against Stockfish (and possibly other engines that were challenging) to see if the specific weaknesses are addressed. Ideally, we'll see an uptick in Scorpion's performance: scenarios that previously led to quick losses are now held as draws or even won. Use a large number of games for statistical significance. Also compare against Phase 7 results to ensure a net improvement. Any remaining outliers (cases where Scorpion still falters) can feed back into another iteration of this data-driven adjustment cycle.

By using real game data in this way, Scorpion continuously learns from stronger opponents – mimicking how human players improve by study and practice. This will guide the engine toward more resilient play and ensure that the theoretical improvements (NNUE, defensive tuning, etc.) translate into practical Elo gains against top competition.

Milestone 5: Benchmarking Progress and Iterative Refinement

Objective: Establish clear metrics and tests to **benchmark Phase 8 improvements**. As we implement the above milestones, we need to track progress quantitatively and ensure each change truly advances Scorpion's goals. This milestone outlines the key performance indicators (KPIs) and how to measure them. By regularly monitoring these metrics, we can iteratively refine Phase 8 features and be confident in the engine's advancement before moving to a next phase.

Key Metrics & How to Measure:

- **Average Centipawn Loss (ACPL):** Measure the average centipawn loss per move in engine games. This is an indicator of move accuracy – it calculates how much the evaluation drops (in hundredths of a pawn) after each of Scorpion's moves [16] [17]. A lower ACPL means Scorpion consistently makes near-optimal moves. *Benchmark:* Play ~100 games of Scorpion vs. a slightly weaker engine or vs. itself, and analyze each move with Stockfish at high depth. Compute Scorpion's ACPL. Phase 8 should show a significant ACPL reduction compared to Phase 7, reflecting fewer mistakes. For reference, top engines have ACPL in the low single digits in many games, while higher ACPL would indicate inaccuracies.

- **Blunder Rate:** Related to ACPL but focusing on major mistakes. Define a *blunder* as any move that swings the evaluation by, say, -100 centipawns or worse (or loses a won game). Count blunders per game on average. We expect Phase 8's defensive improvements to drive this toward zero. *Benchmark:* Using the same analyzed games, count how many blunders Scorpion made (if any). Also track the magnitude of the worst mistakes. Ideally, blunders > 1 pawn should become extremely rare [12]. Comparing blunder counts from Phase 7 vs Phase 8 will directly show if our blunder-reduction strategies are effective.

- **Draw Ratio vs. Strong Engines:** The fraction of games Scorpion draws (versus losses) when playing against stronger engines (like Stockfish or a higher-rated engine). A higher draw ratio indicates

better defensive resilience (the engine can hold equal or slightly worse positions without collapsing). *Benchmark:* Run a match of at least 100 games between Scorpion and a top-tier engine (Stockfish, Lc0, etc.) at a fixed time control. Record the results: how many draws, losses, wins. We expect the draw percentage to increase in Phase 8. For example, if Phase 7 scored 10% draws and 90% losses against Stockfish, perhaps Phase 8 might achieve 30% draws, 65% losses, 5% wins (improvement in not losing). Over many games, even a few more draws is a significant sign of better survival skills. This metric ties into the known trend that as engines get stronger, **their games gravitate toward draws** because they neutralize each other's plans and avoid blunders [12]. Achieving a higher draw rate is thus a positive signal.

- **Position Hold Evaluation Consistency:** This metric examines how steadily Scorpion can maintain a balanced evaluation in drawn or defensive positions over time. In other words, if a position is objectively equal or only slightly worse, does Scorpion's eval stay around that mark as it searches deeper and plays further moves, or does it have late "eval dips" indicating it missed something? *Benchmark:* Set up a suite of difficult defensive positions (fortresses, 3v4 pawn endgames that are theoretical draws, etc.). Let Scorpion analyze each for a long time or play them out against an optimal opponent. Monitor its evaluation over time and outcome. We want to see the eval remain stable (e.g. hovering around 0.00 to +0.20 if drawn) and the game result to be a draw. If the eval suddenly plummets after many moves, that means it failed to hold – possibly a strategic slip. Phase 8's success will be a higher percentage of positions where Scorpion's eval remains consistent and it indeed holds the draw. This can be quantified as, for example, "Scorpion maintained an eval within 0.3 pawns of 0 for 50 moves and achieved the draw" for X% of cases. An improvement here directly reflects stronger long-term defensive planning.

- **Node Efficiency (Nodes per Second & Depth per Node):** Since NNUE can affect speed, we must ensure the engine searches efficiently. **Node efficiency** can be defined as the engine's ability to convert node expansions into effective depth and strength. We look at raw NPS (nodes/second) and effective branching factor. *Benchmark:* Compare Phase 8's NPS to Phase 7 on identical hardware. It may be slightly lower due to NNUE overhead, but should remain within a reasonable range (e.g. $\geq$70-80% of previous speed). More importantly, measure how much Elo gain we get per million nodes searched – if NNUE makes evaluation stronger, Scorpion might achieve higher Elo even with fewer NPS. This can be tested by fixed-depth matches: have both versions search to the same depth or node count per move and see which performs better. A successful Phase 8 will show that despite any speed loss, overall performance is up (meaning each node "does more work" thanks to better eval). We will also profile the search to ensure multi-thread scaling and cluster support still work with NNUE (aligning with Scorpion's parallel search goal [1]). If any inefficiencies are found (e.g. the NNUE eval taking too long or not utilizing SIMD fully), we'll optimize the code or simplify the network architecture to regain speed.

- **Endgame Conversion Rate:** This measures how reliably Scorpion converts winning positions into actual wins, especially in endgames. With improved planning and NNUE, it should avoid throwing away won endgames or getting 50-move-rule draws in theoretically won positions. *Benchmark:* Use a set of known won endgame positions (for example, tablebase positions where one side is winning in X moves). Let Scorpion play them out (or solve them via analysis). Track the percentage of wins converted versus draws/losses from those winning positions. Also track how quickly it finds the winning plan (number of moves or time). Because Scorpion has endgame bitbases up to 6 pieces [2], it should perfectly handle positions once they simplify to those tablebase sizes – Phase 8 must

ensure NNUE integration doesn't interfere with tablebase probing. We'll verify that tablebase wins are still recognized instantly at the threshold. For more complex endgames (7+ pieces, not in tablebase range), a higher conversion rate in Phase 8 means the engine is making better endgame decisions. For example, if Phase 7 sometimes drew King and Queen vs King and Rook due to search horizon issues, Phase 8 should solve that with better understanding (or at least by efficiently using the endgame bitbase if available). An increase in endgame win percentage and reduction of endgame blunders (like stalemating the opponent accidentally) will be an important success criterion.

Benchmarking Process: We will incorporate these metrics into our development cycle: after each major change, run an automated test suite to gather the above stats. Maintain a **progress chart** for each metric across Phase 8 iterations (for instance, track ACPL over time as we integrate NNUE and defensive tweaks – it should trend downward). If a metric stalls or worsens, that flags a regression to investigate. Using statistical significance (SPR tests for Elo, large sample sizes for metrics) is important to ensure improvements are real. By Phase 8's completion, we expect to see **across-the-board improvements**: lower centipawn loss and blunder rates, higher draw rates against top engines, steadier evaluations, reasonable node efficiency, and stronger endgame results. These will collectively confirm that the Phase 8 plan has met its goals in strengthening Scorpion's evaluation and strategic play.

Alignment with Scorpion's Roadmap and Architecture

All the above milestones have been conceived to align with Scorpion's known roadmap and technical design. Scorpion has always been a cutting-edge engine experimenting with neural networks, parallel search, and endgame technology [18] [19]. Phase 8 continues this trajectory by fully embracing NNUE neural eval (advancing the "Neural Networks" goal) and by improving the engine's ability to leverage its **existing features** (like the bitbases and advanced search). For example, integrating NNUE uses the same `egbbdll` framework Scorpion already uses for both NNUE and endgame bitbase probing [5] [2], ensuring we build on the current architecture rather than reinventing it. Defensive play and long-term planning enhancements complement the engine's Monte Carlo Tree Search and Alpha-Beta hybrid approach – a more knowledgeable eval means better node scoring in both search modes, and a solid style means MCTS rollouts are less likely to go astray tactically. In essence, Phase 8 is a natural next step after the previous phase: it shifts the engine from a mix of old eval and new net into a unified, modern engine that is **strong, smart, and stable** in its play. By following this plan, Scorpion will be well-prepared for any future phases (such as further self-learning or specialized tuning) from a robust foundation.

Sources:

- Scorpion (Scorpio) engine features and NNUE integration [13] [5]
- NNUE background and efficiency (Stockfish NNUE example) [3] [7]
- TalkChess/Outskirts engine dev discussions on reducing tactical blunders and search tweaks [20] [10]
- Beuke's engine analysis on draw rates and blunder minimization at top levels [12]
- Definition of centipawn loss and its relation to move quality [16] [17]

---

[1] [2] [5] [13] [14] [15] Scorpio - Chessprogramming wiki
https://www.chessprogramming.org/index.php?title=Scorpio&mobileaction=toggle_view_desktop

3 4 6 7 8 9 beuke.org
https://beuke.org/nnue/

10 11 20 donbot-chess-engine - Outskirts CheSS ForuM
https://outskirts.altervista.org/forum/viewtopic.php?t=5336

12 beuke.org
https://beuke.org/chess-engine-draws/

16 17 Data Science and Chess: Centipawn Loss Elo Correlation | by Enzo Leon Solis Gonzalez | Medium
https://medium.com/@enzo.leon/data-science-and-chess-centipawn-loss-elo-correlation-e06089efd8b8

18 19 GitHub - dshawul/Scorpio: Scorpio chess engine
https://github.com/dshawul/Scorpio