Exploring Emergent Skills with Chess-GPT

Luc Pommeret
Université Paris-Cité, IRIF

Abstract

We explore game strategies in Chess within Large Language Models (LLMs) using games in Portable Game Notation (PGN) from the Lichess database as training data. Our objective is to examine the capacity of LLMs to develop new Skills, which may be considered as a class of emergent properties. We investigate the ability to solve Chess puzzles (games that require a unique correct move) and seek to assess the success rate of an LLM in performing this task. Subsequently, we study how this success rate changes when we alter the **elo** parameter context in the header of the PGN.

1 Introduction

The exploration of Chess within the realm of artificial intelligence offers a uniquely fertile ground for probing the capabilities of Large Language Models (LLMs) beyond conventional game strategies. Unlike traditional Chess engines that focus solely on identifying the optimal move, LLMs can employ a wealth of historical game data in PGN notation to craft nuanced strategies that mirror human play. In this paper, we utilise PGN notation not just as a record

```
[White "Garry Kasparov"]
[Black "Magnus Carlsen"]
[Result "1/2-1/2"]
[WhiteElo "2900"]
[BlackElo "2800"]

1. e4 e5
2. Nf3 ...
```

Figure 1: Chess match between Garry Kasparov and Magnus Carlsen

of past games but as a dynamic tool to guide the strategic outputs of LLMs. By manipulating PGN headers—such as Elo ratings, game outcomes, and player details—we can direct the model's behaviour in solving Chess puzzles or emulating specific strategic styles, as illustrated by the example of a history of a Chess game in `PGN` format between Garry Kasparov and Magnus Carlsen (see figure 1). This manipulation showcases the flexibility of LLMs to adopt diverse and human-like strategies, allowing for an in-depth analysis of emergent behaviours.

Emergent properties in this context result from the collective interactions among the model's components, which transcend the capabilities of any single part. We will review various definitions of an emergent property and study the situation in Chess. The ability to adjust strategies based on subtle changes in game context further illustrates a level of robustness comparable to that of human players, who adeptly adapt their strategies in new situations.We study the property of solving puzzles when we modify the **elo** parameter.

This paper examines the impact of adjustments to the `PGN` headers on the decision-making processes of large language models (LLMs), influencing their playing styles accordingly. It delves into the concept of emergent properties, emphasising how LLMs develop capabilities that were not explicitly taught but emerge from their training environment and inputs. Particular attention is given to the task of puzzle-solving, defined as the ability to identify the correct move in a given position with high probability.

The subsequent sections of this paper will explore these emergent properties in detail, using Chess-based examples to demonstrate how subtle manipulations in training data can significantly alter LLM behaviour. This exploration not only enhances our understanding of LLMs within the Chess domain but also illuminates their broader potential in strategic applications, marking a significant shift from pursuing optimal solutions to engaging with complex problems in a more human-like manner.

Section 2 explores the nature of emergent properties, Section 3 introduces probing techniques for LLMs, and Section 4 outlines methods to quantify the ability to solve puzzles and detect emergent properties relative to the **elo** parameter.

2 What is an Emergent Property?

Emergent properties arise from the collective interactions of a system's components, beyond the properties of the individual parts. *Phase transitions* in random structures with a parameter p illustrate this concept. Consider random graphs following the Erdös-Renyi model [6], where p is the probability to select independently one of the $\binom{n}{2}$ simple symmetric edges. With high probability, if $p < \log n/n$ the graph is not connected, whereas if $p \geq \log n/n$ the graph is connected. The value $p = \log n/n$ is the phase transition for the connectivity property. In Physics, the parameter p is often related to the temperature, where similar phase transitions exist.

In the realm of large language models (LLMs), emergent properties can be seen as the manifestation of Skills that exceed the explicit content of the training data. Arora discusses how these are Skills developed from the latent potentials embedded within the training [2]. Is it possible that the training data combine groups of up to 3 different Skills, and that the LLM's answers witness many more Skills, up to 6 different Skills for example? Arora's approach, based on combinatorial considerations shows that is is possible. The LLM's answers witness Skills that did not occur in the training data, hence are emergent.

This is further explored in the context of both Natural Language Processing and strategic games. In the case of Natural Language, one may want to learn the syntactic structure of a sentence, from the `Word2Vec` vectors of the words of the sentence [7]. In games [9, 4], the language is the sequence of runs (for example b2, f4,.... in the `PGN` notation for Chess or Othello) (see figure 2). The training data are taken from Lichess data base [1] which store the history of games, and the LLM learns the function next which provides the next move in a run. Can the Neural Network describing the function next also predicts the position of each piece of the game, *i.e.* the World Model? If it is possible, the authors say the the position of the game is an emergent property, as no rule of the game was ever given. Nicholas Carlini uses the term "Language Modeling; Not Winning" to describe how LLMs may also learn not to win, but to play like humans [4]. They obtain these results by considering probes, we now describe.

3 Probing to Detect Emergent Properties

Probing techniques [3] are used to uncover latent structures in deep neural networks, based on some internal states of the network. Assume an input x of dimension d in a network which computes $f(x)$, and let x^i be the vector of values at layer i of the network. Probing is reading some of the x^i and learning a new network as a classifier for a new property (a function g), from the same training data. We don't compute relative to f, but relative to some states of the algorithm which computes f.

In NLP, [7] considers a sentence as an input x which is a sequence of n words given to a Neural Network which computes some function f. Internally, it computes the `Word2Vec` representation of the words w_i as vectors v_i of low dimension. We can learn a matrix B which transforms the vectors into $v'_i = B.v_i$ such that the new norm $(B;v)^t.(B.v)$ defines a new distance dist_B. The function g is a classifier for the property: dist_B is close to the distance associated with a syntactic tree of the original sentence. They probe the original network, and the new property g is emergent.

In games, a natural question is to ask if we can approximate the function $\mathsf{Position}(i,j) = k$ which describes the piece on the board in position (i,j) from a subset x^i of the internal layers of the Network. In the Othello game [9], $k \in \{\perp, \text{black}, \text{white}\}$ whereas in Chess [4, 8], the value k also describes the type of each piece as in section 4.1. If the function $\mathsf{Position}(i,j) = k$ can be approximated by probes on a subset of the the Neural Network's layers defining the function next, we say that the *World Model* is an emergent property.

3.1 Intervention on probes

If we approximate the Position function, we can try to modify the board, by removing a piece in Chess [8] or by switching a black piece for a white piece in Othello [9]. These modifications are called *Board interventions*. We can then infer an $(x')^i$ by Gradient descent on the Network defining the probe and compute a new value for the function $f(x)$, where the values $(x')^i$ replace the values x^i. We can then verify if the prediction next is coherent with the intervention. Typically Heatmaps [9] give the most likely moves and should adapt to the intervention. Interventions can also concern Skill levels (the **elo** grade in Chess) in [8].

4 Puzzle Solving Skills

4.1 Definitions for Puzzle Solving Skills

We delve into the puzzle-solving prowess of Large Language Models (LLMs) such as Chess-GPT, evaluating their capacity to utilise and adapt learned strategies to novel and unstructured challenges. This aptitude underscores their emergent cognitive abilities and serves as a metric for assessing the depth and flexibility of their learned behaviours. Let us acquaint ourselves with the following terms:

- **PGN (Portable Game Notation)**: A data structure that includes a *header* detailing the players, their rankings, the game outcome [1-0], [0-1], or [1/2-1/2], followed by a sequence of moves (see figure 1).
- **Elo rating** is a number assigned to a player or puzzle indicating its level of difficulty. For a precise definition, we refer to Wikipedia. This definition is implemented on Lichess.com [1].

The allure of Chess puzzles lies in their unique solutions: these puzzles are positions extracted from actual Chess games, each demanding a specific correct move.

Definition 1 *$\boldsymbol{M}$ is the set of possible moves:*

$$\boldsymbol{M} = \{(p, i, j) : p \in \textit{Chess pieces}, i \in [\![a; h]\!], j \in [\![1; 8]\!]\}$$

where the set of Chess pieces is $\{\emptyset, R, N, B, Q, K\}$.

Definition 2 *$\texttt{puzzle\_PGN}_e$ is the set of pairs (x, y) such that $x \in$ PGN and $y \in \boldsymbol{M}$ is the unique correct move at x, e is the **elo** level of the puzzle as given in the Lichess database [1].*

$$\texttt{puzzles\_PGN}_e = \{(x, y) : x \in \textit{PGN}, y \in \boldsymbol{M}, e \in \textbf{\textit{elo}}\}$$

*where **elo** is the interval $[\![1000; 3000]\!]$.*

We fix an **elo** level e and select $(x, y) \in \texttt{puzzles\_PGN}_e$ from the Lichess database with a uniform distribution.

Definition 3 *An LLM possesses the puzzle-solving Skill for an **elo** level e and at rate α if*

$$\mathsf{Prob}_x[\mathsf{next}(x) = y \mid (x, y) \in \textit{puzzles\_PGN}_e] \geq \alpha.$$

This methodology enables us to precisely evaluate a specific Skill: the ability to identify (with great probability) the best move given a position. It involves providing the LLM with a PGN game history (refer to figure 1) and querying it for the next token (in this case, the next move).

Our focus here is the rate of Skill for a particular class of puzzles and a fixed **elo** level. In this context, an intervention is the modification of the **elo** parameter. We then study how the Skill rate increases if the **elo** number increases and decreases if the **elo** number decreases. This is the emergent property we are trying to test.

Karvonen [8] has demonstrated that when the LLM is ”primed” (in the prompt) with a score [0-1] and made to play as white, its performance deteriorates compared to being primed with [1-0].

4.2 Probing Techniques and Noise Impact in Puzzle Solving

The *probing techniques* (as described earlier) will be invaluable for deeply understanding how the LLM constructs a *World Model* of the puzzle. We will also employ *interventions* to study the stability of puzzle resolution through the *addition*, *substraction*, or *relocation* of pieces.

In examining the *stability* of puzzles, we are effectively exploring the *robustness* of LLMs to changes in the position. LLMs are known to exhibit human-like behaviour. One such characteristic is robustness to situational changes: *a minor modification in the puzzle should lead to an adaptation*. We must quantify it.

We also wish to investigate the effect of noise in the training data, similar to Alexandros Dimakis' study in the domain of images [5]. For this purpose, we have implemented an algorithm that utilizes `Stockfish` and incorporates a feature where each move has a 5% chance of being completely random (yet legal) to generate training data. Our primary aim is to verify how this translates into the puzzle resolution rate, as well as into the *World Model* and the resultant heat maps.

5 Conclusion

The study of games in AI, and in particular for Chess, is directed on winning strategies. An LLM is not a generator of winning strategies. It allows for a more refined management of the Chess behaviour we wish to adopt, which leads us to:

1. Study its *World Model* to more precisely understand its perception of a specific Chess position.

2. Conduct statistics on its success rate (in relation to the **elo** parameter) in solving puzzles (which is made possible by the uniqueness of the correct move in the puzzle).

3. Study the robustness relative to interventions on the gameboard and use fine-tuning to improve the success-rate.

4. Later, examine the influence of statistical noise on learning, by having it learn games in which each move has a 5% chance of being random.

References

[1] Lichess: Free online chess. https://lichess.org/. Accessed on: may 4 2024.

[2] S. Arora and A. Goyal. Skill-Mix: A Flexible and Expandable Family of Evaluations for AI Models, 2023.

[3] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48:1–12, 2021.

[4] N. Carlini. Playing chess with large language models. https://nicholas.carlini.com/writing/2023/chess-llm.html, 2023.

[5] Giannis Daras, Kulin Shah, Yuval Dagan, Aravind Gollakota, Alexandros G. Dimakis, and Adam Klivans. Ambient diffusion: Learning clean distributions from corrupted data, 2023.

[6] P. Erdös and A Renyi. On the evolution of random graphs. In *Publication of the mathematical institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.

[7] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019.

[8] A. Karvonen. Emergent World Models and Latent Variable Estimation in Chess-Playing Language Models, 2024.

[9] K. Li, F. Viegas, A. K. Hopkins, H. Pfister, D. Bau, and M. Wattenberg. Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

Appendix

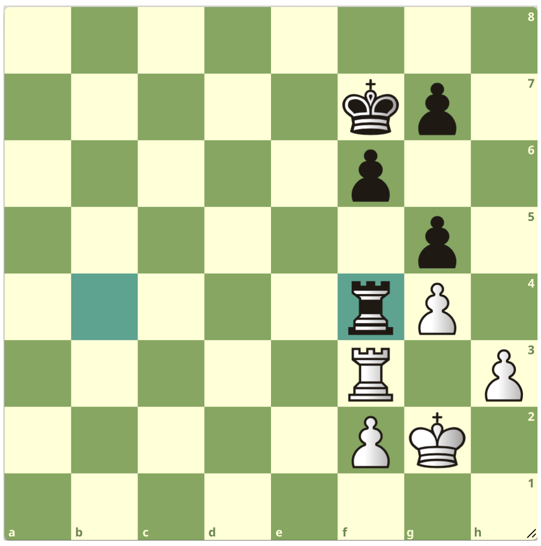

Figure 2: Chess position after 37...Rf4 : the good move is Rxf4

Example of a data structure in PGN format:

```
[Event "Simulated Game"]
   [Site "Local Simulation"]
   [Date "2024.05.02"]
   [Round "?"]
   [White "Player1 (1800)"]
   [Black "Player2 (1800)"]
   [Result "1-0"]
   1. Nf3 Nf6
   2. h3 e6
   3. e3 d5
   4. d4 c5
   5. c4 dxc4
   6. Nbd2 b5
   7. b3 a6
   8. bxc4 Bb7
   9. dxc5 bxc4
   10. Rb1 Ra7
   11. Nxc4 Qxd1+
   12. Kxd1 Ne4
```

```
13. Rb2 g6
14. Nd4 Nd7
15. c6 Bxc6
16. Nxc6 Nc3+
17. Kc2 Ne4
18. Nxa7 Bg7
19. Rb7 g5
20. Nc6 h5
21. a3 Ndc5
22. Ra7 Rh6
23. Bd2 Rf6
24. N4e5 g4
25. hxg4 Nd6
26. Ra8+ Nc8
27. f3 Nb7
28. Rxc8+ Nd8
29. Bxa6 Rh6
30. Rxd8 1-0
```