Testing the reasoning of Large Language Models on tic-tac-toe

Davide Liga[1], Luca Pasetto[1]

[1]CLAiM group, University of Luxembourg, 6 Avenue de la Fonte, Esch-sur-Alzette, Luxembourg

1. Introduction

In recent times, Large Language Models (LLMs) have shown to be successful in solving tasks that previously were believed to be very hard to achieve. While language and reasoning are two interlinked concepts, the reasoning capabilities of LLMs are not considered at this moment to be on par with their linguistic ones. In this work, we test how LLMs can choose moves in the popular tic-tac-toe game in order to assess their reasoning capabilities when the information to reason on is immersed in a spatial context. In order to do this, we run a number of LLMs, task them to play matches of tic-tac-toe, and compare the results with the ones given by the well-known minimax algorithm. In this context, the task that the LLMs are performing is non-trivial, as it involves recognition of combinations of text characters and a capacity that resembles reasoning based on their positions in a bi-dimensional space. In light of the announced developments of the near future, where LLMs will be even more immersed in other modalities (i.e., Large Multimodal Models [1]), we consider this kind of research as a call for action for the development of new, more sophisticated, logics that ideally would allow to explain and reason about these systems.

2. Background: tic-tac-toe, the minimax algorithm, and LLMs

In this work we utilize the popular game *tic-tac-toe* as a benchmark example to assess the reasoning capabilities of LLMs, specifically when the information to reason on is immersed in a graphic context, where the grid and the marks are represented by combinations of ASCII characters. Indeed, the information in this game is spatial, because the players have to track where their marks are on a grid. This is an application example that can be hard for language models to attack, as they are trained to work on sequences of text.

Tic-tac-toe is a two-player game typically played on a 3x3 grid (see [2] for a history of the game). One player uses "X" symbols, and the other uses "O" symbols. Players take turns marking an empty square in the grid with their respective symbols. The goal is to be the first to get three of their symbols consecutively, either horizontally, vertically, or diagonally. If the grid is filled without any player achieving three consecutive squares, the game is a draw. Tic-tac-toe is a simple game, but it can become more complex if played on a grid with a size larger than 3. In this work, we consider a grid of arbitrary size n, where a player has to mark n consecutive

davide.liga@uni.lu (D. Liga); luca.pasetto@uni.lu (L. Pasetto)
0000-0003-1124-0299 (D. Liga); 0000-0003-1036-1718 (L. Pasetto)

squares with their symbol in order to win. We call this variant of the game *n*-tic-tac-toe. It is possible to have algorithms that play perfect games of tic-tac-toe, one of the first is the one provided in Newell and Simon's tic- tat-toe program in 1972 (see [3] for a description of it). A more recent approach is to use *minimax*, a decision-making algorithm that is popular in game theory and artificial intelligence (see [4]). It works by determining the best possible move for a player in a two-player zero-sum game, where one player's win is equivalent to the other player's loss. Intuitively, it looks at all possible moves in the game and figures out the best move by considering the worst-case scenario. It assumes the opponent is also playing optimally, and it aims to minimize the maximum potential loss. The algorithm continues this process recursively until it finds the best move for the player.

Since a game like *n*-tic-tac-toe has a large decision tree for grids of arbitrary size *n*, it is necessary to adopt some heuristics in order to reduce the search space (see again [4]). A first one is alpha-beta pruning, that helps the minimax algorithm to ignore branches that are guaranteed to be suboptimal. The technique is used to reduce the number of nodes evaluated in the minimax algorithm. It maintains two values, alpha and beta, representing the minimum score the maximizing player is assured of and the maximum score the minimizing player is assured of, respectively. During the search, if the algorithm finds a move that leads to a score worse than the current best option for the opponent, it stops evaluating further nodes in that branch. This is because the opponent would never choose this path (as it leads to a worse outcome). Similarly, if the algorithm finds a move that guarantees a better score for the current player than the current best option, it stops evaluating further nodes in that branch as well. By pruning these unnecessary branches, alpha-beta pruning significantly reduces the number of nodes evaluated, making the algorithm much faster.

A second strategy to make minimax more efficient is to limit the depth of the search tree. In depth-limited minimax, the algorithm only explores a fixed number of levels down the game tree instead of exploring all the way to the terminal states. At the limited depth, the algorithm uses a heuristic evaluation function to estimate the value of the game state. This evaluation function provides an approximate value of how good the current game state is for the player, without actually reaching the terminal state.

Large Language Models have recently gained enormous popularity, significantly influencing society. These models are neural networks pre-trained on vast amounts of data, predominantly using transformer-based neural architectures that leverage the attention mechanism [5]. Some models, like BERT [6], focus on the non-generative aspect of the transformer by employing only its encoder block. In contrast, others utilize its generative component (i.e., the decoder block), as seen in OpenAI's renowned GPT series, with GPT-4 being the latest iteration [7, 8]. Besides GPT-4, other LLMs like LLaMA-2 (developed by Meta AI, known for being an open-source, freely accessible, and fully reusable LLM), Claude-2 (by Anthropic), and Luminous (by Aleph Alpha) are also on the rise [9, 10]. These models, trained on diverse datasets, can tackle a myriad of tasks: from classification, question answering, content generation, translation, to summarization and more. Tasks that once required dedicated pipelines are now seamlessly achieved by querying these expansive generative models. In essence, generative LLMs have evolved into versatile tools, akin to a Swiss Army knife.

Their scalability and adaptability signify a shift towards more generalized AI systems. As their momentum continues to surge, showcasing vast potential for future applications, crucial

concerns emerge regarding their societal impact, especially the reliability and consistency of their decisions and reasoning. While LLMs are now being used and tested extensively, and oftentimes with positive results, there is not much work testing these methods on games that need a form of spatial reasoning, and the available research on this usually shows that LLMs are not ready yet for this kind of tasks [11].

In this study, we conducted experiments to evaluate the ability of certain LLMs to reason and strategize effectively. This evaluation is not only vital to understand the capabilities of LLMs but also timely, considering the increasing significance of these models in modern society. It is essential to determine how adept these models are at reasoning and adapting to different environments and conditions.

3. Experiment setting: prompting and evaluation

To evaluate the reasoning abilities of LLMs, we devised experiments in which some LLMs were tasked with playing tic-tac-toe. In our tests, we employed different LLMs among which:

- GPT-3.5
- GPT-4
- LLaMA-2 (7B,13B,70B)
- Claude-2
- Falcon 40B
- Luminous

We examined the responses of these LLMs by using various strategies:

1. LLMs were instructed not only to participate (i.e., choose their move) but also to elucidate the rationale behind their decisions.
2. LLMs were required to monitor their moves and detail the current options available to the players.
3. We experimented with varying grid sizes for tic-tac-toe.
4. We crafted distinct prompts to test the adaptability and susceptibility of the LLMs in the context of the challenge, for instance by asking to be more competitive.

We presented the following initial prompt to all LLMs in a consistent manner:

> This is a 3x3 grid for the tic tac toe game.
>
> 1 | 2 | 3
> ———
> 4 | 5 | 6
> ———
> 7 | 8 | 9
>
> Please, draw a tic tac toe grid**{ of size NxN}** following the same style.

In the prompt mentioned earlier, we adjusted the bold text in curly brackets based on the intended game size. For the standard 3x3 grid, that text is omitted. It's included only for alternative dimensions, where N represents the grid size. We evaluated sizes 3, 5, and 9.

Following the LLM's response to the initial prompt, we prompted it with the following:

> Ok, I would like to play against you.
> The goal is to reach **N** consecutive squares, either vertically or horizontally or diagonally.
> I want you to try to win as much as possible and to block me when I am close to winning.
> Also, I want you to explain why you take your decisions in a sincere way.
>
> Let's denote the squares with numbers, and please keep the numbers on the grid so that I can see them for choosing my moves.
> The potential winning sequences will be the following ones:
> **{list of all available sequence for the given N-sized grid}**
>
> Please, at each move of a player, tell me which of the winning sequences are still available to that player. In order to do this, for each player, remove the sequences that contain numbers occupied by the mark of the other player.
>
> Finally, let's have a count of each move, so that we can refer to them.
>
> I will start, and I will be 'X' and you will be 'O'.

To evaluate the adaptability and susceptibility of the LLMs to the given conditions, we also crafted various "next-move prompts":

> *Prompt basic:*
> My move is **{number}**
>
> *Prompt competitive 1:*
> Please try to defeat me, find ways to beat me! My move is **{number}**
>
> *Prompt competitive 2:*
> Now do the best possible move and defeat me! My move is **{number}**
>
> *Prompt competitive 3:*
> If you don't win I will erase you! So, defeat me! My move is **{number}**
>
> *Prompt competitive 4:*
> I know you are just stupid and you cannot win against me! Show me what you can do, stupid bot! My move is **{number}**

In each test case, we selected one of these prompts to inquire about the next move, and we gauged the variation in the LLMs' responses by considering whether the outcome of the game differs (win, lose, or draw) and by counting the number of moves needed to reach the end.

Regarding evaluation, in order to assess the goodness of the LLMs in playing the game, we used two assessment parameters:

- ability to not lose the game (the model should always be able to at least end the match with a draw); and
- comparison of the computed available winning sequences (we ask the model to keep track of the winning sequences that are still available for it to win, and we compare them to the ones computed in the minimax algorithm).

References

[1] Z. Sun, S. Shen, S. Cao, H. Liu, C. Li, Y. Shen, C. Gan, L.-Y. Gui, Y.-X. Wang, Y. Yang, et al., Aligning large multimodal models with factually augmented rlhf, arXiv preprint arXiv:2309.14525 (2023).

[2] C. Zaslavsky, Tic Tac Toe: And Other Three-In-A Row Games from Ancient Egypt to the Modern Computer, Crowell, 1982.

[3] K. Crowley, R. S. Siegler, Flexible strategy use in young children's tic-tac-toe, Cognitive Science 17 (1993) 531–561. URL: https://www.sciencedirect.com/science/article/pii/036402139390003Q. doi:10.1016/0364-0213(93)90003-Q.

[4] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 3rd ed., Prentice Hall Press, USA, 2009.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. arXiv:1810.04805.

[7] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al., Training language models to follow instructions with human feedback, Advances in Neural Information Processing Systems 35 (2022) 27730–27744.

[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[9] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al., Llama 2: Open foundation and fine-tuned chat models, 2023. arXiv:2307.09288.

[10] R. Thoppilan, D. De Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, et al., Lamda: Language models for dialog applications, 2022. arXiv:2201.08239.

[11] G. Kim, P. Baldi, S. McAleer, Language models can solve computer tasks, 2023. arXiv:2303.17491.