Universiteit Leiden

# Master Computer Science

Exploring LLM intelligence using games


| | |
|---|---|
| Name: | Stijn Boere |
| Student ID: | 1831968 |
| Date: | 18/08/2025 |
| Specialisation: | Artificial Intelligence |
| 1st supervisor: | Dr. Mike Preuss |
| 2nd supervisor: | Giulio Barbero |


Master's Thesis in Computer Science


Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Einsteinweg 55
2333 CC Leiden
The Netherlands

# Exploring LLM intelligence using games

Boere, S., Barbero, G., Preuss, M.

LIACS, Leiden University,
Einsteinweg 55, Leiden

**Keywords:** LLMs, intelligence, Game AI

**Abstract.** Since the release of ChatGPT in late 2022, large language models (LLMs) have been gaining popularity due to their ability to deal with complex problems. While the consensus of the general public is that LLMs are just as, if not smarter than their users, experts argue that LLMs are incapable of intelligence and warn for their impact. In this paper, we examine the behavior of LLMs and investigate if they can be considered intelligent. We employ multiple games as testbeds to observe the LLM's behavior. We let our choice of subsequent games be informed by the behavior shown in prior games. We first explored chess, a baseline in the field of AI for decades. Based on the obtained results, we explored checkers followed by tic-tac-toe. Given the performance of the LLM in these board games, we finally opted for the text-based game Zork. The LLM exhibits a multitude of problems within these games, and most prominently shows a lack of reasoning abilities, which can be tied to known LLM shortcomings. This leads us to believe that the behavior displayed by the LLM can not be considered intelligent.

## 1 INTRODUCTION

All throughout history, advancements and inventions have heralded in new revolutions and eras, forever changing how humans live their lives. The First Industrial Revolution (late 1700s - early 1800s), succeeding the Second Agricultural Revolution, used steam and water power to transition hand-production methods into machine-powered production methods (Deane, 1979)(Liao et al., 2018). The Second Industrial Revolution (late 1800s - early 1900s) brought us extensive railroad and telegraph networks, allowing electricity, people and ideas to flow faster and more freely around the world (Mokyr and Strotz, 1998). The Third Industrial Revolution (late 1900s) brought us the widespread digitization of communication and industrial processes via personal computers and the Internet (Roberts, 2015). There is debate around whether the developments in the field of artificial intelligence (AI) herald us into the Fourth Industrial Revolution (Babu et al., 2024), or whether AI is the culmination of the Third Industrial Revolution. One thing is for certain: we are currently at the dawn of the AI era.

AI has been around for a decades, but it wasn't until the current AI boom (2013)(Miyazaki and Sato, 2018) that AI was as extensively applied and used to the extent that it is today. The AI boom brought with it generative AI technologies, including large language models (LLMs) and AI image generators. LLMs quickly took the world by storm, in particular the Chat Generative Pre-trained Transformer (ChatGPT) which reached an estimated amount of 100 million users just two months after its launch in early 2023 (Milmo, 2023).

LLMs are enabled by multiple technologies, such as Big Data, AI, Cloud Computing, Edge Computing and Human Computer Interaction (Yenduri et al., 2023). In turn LLMs, and in particular the Generative Pre-trained Transformer (GPT) models, have had a significant impact on several applications, such as education, healthcare, agriculture and more (Alhafni et al., 2024)(Haltaufderheide and Ranisch, 2024)(Shaikh et al., 2025). However, not only industries have been affected. Individuals have also started relying on these models for their daily tasks, helping them with their scheduling, email drafting and information gathering (Shi, 2024). Students in particular have been a large consumer of the service, with numbers as high as roughly 90% of students using ChatGPT in their schoolwork (Ettinghausen, 2025). Experts have voiced their concerns stating that ChatGPT usage might have a negative impact on students' higher-order thinking skills (Klopfer et al., 2024), assessment practices, while opening up to new problems related to scientific integrity (Farhi et al., 2023).

LLMs have many known shortcomings (Kucharavy, 2024), yet many people are not equipped with the knowledge to notice these failures, and instead blindly rely on the output of these

models. This begs the question: How reliable are these models really? And are these models actually intelligent? In this research, we use games as testbeds to observe LLMs' behavior. In doing so we hope to answer the following question: "How do LLMs behave in different games, and can they be considered intelligent?".

### 1.1 Overview

We will first discuss the history of AI in games, and emphasize the usefulness of games in research, to set the stage for LLMs. We will then describe LLMs, their known problems and provide a definition of intelligence. Equipped with these principles, we will then describe a separate methodology for each of the games we employed. Each of these methodologies then ties into the subsequent results. The patterns as observed within these results are then analyzed and compared to the earlier defined known LLM problems as well as the definition of intelligence to provide an answer to our research question. Finally, our research, its shortcomings and its implications for future research are discussed.

## 2 RELATED WORK

### 2.1 Game AI

Games have long been an emblematic medium for AI development, either as challenges, experimental grounds or opportunities for actual commercial applications.

**Deep Blue** In 1997, after a 6 game match IBM's Deep Blue chess computer (Campbell et al., 2002) reigned victorious against Garry Kasparov, the then-reigning world champion in chess. It was the first to win a game, or even a match against a figure of that stature under regular time controls. Prior to this win, chess proficiency had been a goal for AI developers for decades (Iqbal, 2010). Following this win, games of greater complexity, such as Go, have replaced chess as the new challenge. What seemed like an impossible feat decades prior quickly turned into a milestone in the ever-evolving field of AI. Even decades after the struggle between man and machine, we can see the influence that Deep Blue had in fields like philosophy, artificial intelligence and computer science (Hoekenga, 2007).

**AlphaGo** While Deep Blue was able to conquer chess using a brute force-based computational method, this approach could not be scaled to Go due to the vastly greater complexity of the game. Google DeepMind's AlphaGo model (Silver et al., 2016) attempted to solve this by using deep neural networks with advanced search algorithms, namely MCTS. Using this approach, they successfully beat the European Go champion five to zero in 2016, and in 2017 won against the number one Go player in the world in a three-game match, after which AlphaGo was retired. AlphaGo is succeeded by AlphaZero (Silver et al., 2017), which has a 100% win-rate against the former. The success of AlphaGo and AlphaZero further solidified the strength of the underlying neural networks in complex domains, as well as how the usage of reinforcement learning allowed the solving of harder problems through trial-and-error.

**AlphaStar** StarCraft was heralded by some scientists at DeepMind as the next target after Go would be beaten. After AlphaGo succeeded in doing exactly this in 2016, DeepMind introduced AlphaStar (Arulkumaran et al., 2019) in 2019 as the first AI system to beat a professional player at StarCraft II, a game which is arguably harder than Go due to several factors such as the real-time play, partial observability and the complexity of the rules. The success of AlphaStar has been argued by scientists at DeepMind to benefit systems that need to operate on partially observable information, such as self-driving vehicles or robots.

**Hide and Seek** Similarly to how the Alpha models by DeepMind, using reinforcement learning, were able to train using self-play, OpenAI was able to train agents by pitting them against themselves in a game of Hide and Seek (Baker et al., 2020). The agents playing this game of Hide and Seek show several emergent strategies. While initially the hiders simply run around trying not to get caught, eventually they interact with boxes present in the level as shown in Figure 1. These boxes are then used to block the entrance to a present structure, essentially leaving the seekers unable to reach the hiders. Through this adversarial play, the seekers eventually develop the strategy of taking a ramp to scale the walls and enter the structure in which the hiders barricaded themselves. In response to this, the hiders learn to take the ramp with them into the structure, leaving the seekers unable to scale the walls. This back and forth between hider and seeker is emergent behavior from its self-play: no other external factors influence the agents to move in this fashion.

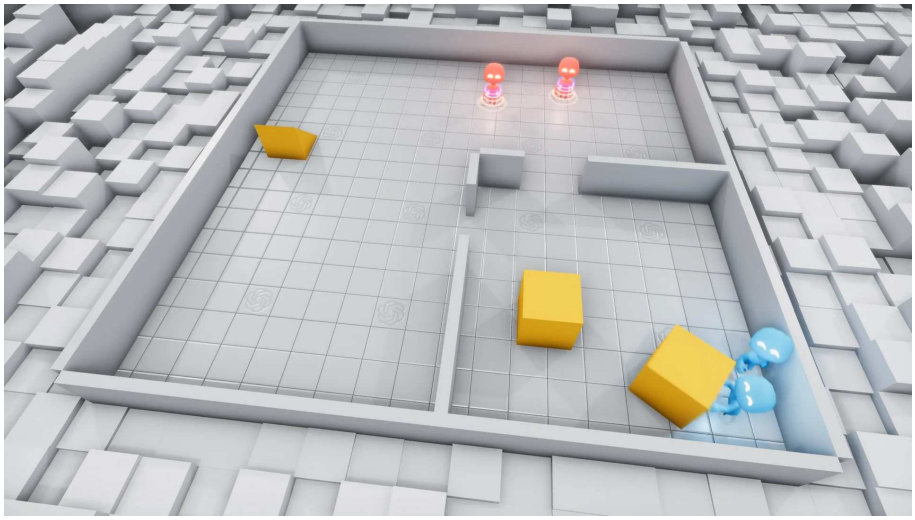

Fig. 1: Shown is a sample OpenAI Hide and Seek environment in which the hiders (blue) have to avoid capture from the seekers (red). This environment contains a structure with two doorways, a ramp and two boxes. Interactable items are colored yellow and can be moved or locked by the hiders to prevent the seekers from doing this themselves. Image courtesy of (Powell et al., )

**Video games for research** Several properties of games allow them to find different purposes in research (Gómez-Maureira et al., 2022). Games are a great tool for invoking a measurable emotion or reaction in the player. Games used to achieve this, are used with the purpose of it being a stimulus. A game being used as a stimulus is supposed to evoke a short-term reaction (Ferreira et al., 2023). Should the goal be to change the behavior of the player (for their own benefit) on the long-term, the games can be used with the purpose of intervention. Yet another purpose of games is their incentive. Undesirable tasks can be made more appealing when presented in the context of a game (Deterding et al., 2011). Finally, games can be used to model. Here, a process in the game is the subject of study. This process can be on a conceptual level or this process can be an as accurate as possible representation of another topic of interest. Contrary to the previous three, the focus of modeling mostly lies in the system rather than the player.

### 2.2 LLMs

Transformers, a neural network structure as introduced in the "Attention is all you need" paper (Vaswani et al., 2023), form the building blocks for LLMs. With the transformer comes the introduction of the attention mechanism. This mechanism allows the model to weigh the importance of parts of the input sequence, which allows the model to capture the relevance between different parts in this input sequence, even if these parts are far apart.

Transformers consist of two parts: an encoder part and a decoder part as shown in Figure 2. The encoder part takes an input and creates an embedding from this input. Rather than simply creating embeddings on word-level, essentially creating an embedding per word, the transformer has a fixed output size representative of the entire input sequence. This means that an input of one word has the same sized embedding as an input the size of an entire document. The transformer is thus trained to not simply embed words, but actually embed the meaning behind and, using the attention mechanism, between words and sentences. Using these embeddings we can then compare inputs, be it words, sentences or documents, to each other based on the underlying meaning behind the text rather than the words themselves used in the text. This is one of the use-case examples of LLMs which only use the encoder part of the transformer. A prominent encoder-only LLM is the Bidirectional Encoder Representations from Transformers(BERT) (Devlin et al., 2019) used for tasks such as question answering and language interference.

The decoder part of the transformer takes input and, based on this input, predicts an output. More specifically, it outputs a list of words, with each word having a probability of it being the most likely output given the input. Using the attention mechanism allows the model to refer to previously mentioned words in its output, creating believable human-like text.

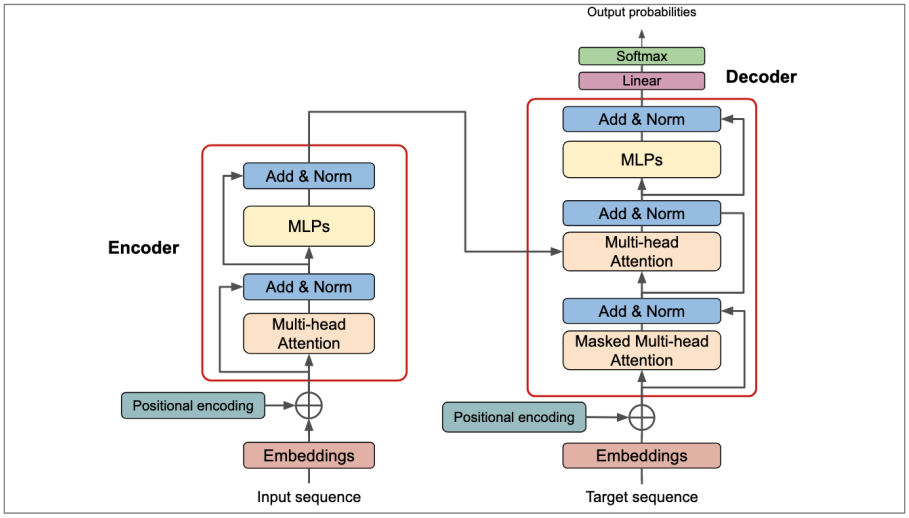


Fig. 2: The transformer architecture, adapted from (Vaswani et al., 2023) by (Nyandwi, 2023). The left side is the encoder part of the transformer, whereas the right side is the decoder part. Both parts contain their own attention segment.

Each output is appended to the input and then again used to predict the next output. This essentially iteratively builds a sentence from scratch by simply predicting the next word in the sequence. A model that uses the decoder-only transformers is the Generative Pre-trained Transformer (GPT) model (Radford et al., 2018), popularized by the ChatGPT framework.

### 2.3 LLM intelligence

Given their excellence at seemingly human-like speech, there is a debate around whether or not LLMs are in fact intelligent. The vast majority of the population seems to think that the LLMs they use are just as smart, if not smarter than themselves (Bureau, 2025). This belief is further solidified by the fact that LLMs were recently able to pass the Turing test (Jones and Bergen, 2025), a test that has long been used as a metric for machine intelligence. However, experts argue that the test is outdated, or not representative of true intelligence, given that it focuses more on the imitation and tricking of humans rather than an actual measure of intelligence (Churchland and Churchland, 1990). While LLMs perform seemingly well, they don't come without their shortcomings (Kucharavy, 2024):

**Generative LLMs cannot be factual** Given the probabilistic manner in which LLMs generate their responses, a prompt that is new to the LLM will be based on what the LLM deems a plausible response, rather than grounded in facts. If a prompt was part of the training data, and seen enough during this training process, the LLM might generate an item memorized from its training dataset. But for the vast majority of unseen prompts, this is not the case.

**LLMs with auxiliary tools still struggle to be factual** External databases can be used by LLMs to provide factual information. However, this is not a process without its own problems (Glaese et al., 2022)(Shuster et al., 2022). Firstly, the LLM needs to understand the information need of the user. Secondly, the LLM needs to generate a proper query for the external tool. Thirdly, there is a chance that the factuality of the information provided by the external tool might be compromised. Finally, the LLM needs to be able to follow instructions and summarize the information provided by the external tool.

**Generative LLMs will leak private information** Certain LLM-families have been shown to be particularly good at their memorization capabilities. Models like ChatGPT have been able to recall personal information such as names, phone numbers, email addresses, bank information and similar items (Carlini et al., 2021). It should thus not be assumed that these items remain private between the LLM and the current user.

**Generative LLMs have trouble with reasoning** LLMs have been trained to generate the most probable response to the prompt based on similar prompts in the dataset. As such their reasoning abilities do not exceed what they've encountered multiple times in this training dataset. This means that they are able to perform a simple arithmetic operation and perform basic reasoning (Brown et al., 2020), yet they do not have inherent reasoning abilities.

**Generative LLMs forget fast and have a short attention span** The attention span of LLMs is limited by the amount of tokens they can process at once. Models like GPT-3 have a span of around 2 pages, meaning it has trouble referencing and remembering text prior to this.

**Generative LLMs are only aware of what they saw in training** Given that LLMs learn to respond to prompts seen in the training set, their ability to respond to prompts too dissimilar from anything they've seen in the training set drops significantly when presented with such. This pertains to things like current events, or subjects too niche to have been properly represented in the training set. A workaround to this problem is the utilization of external tools, which introduces problems in itself.

**Generative LLMs can generate highly inappropriate texts** Larger LLM models are trained on a vast amount of training data. This data can be obtained from various sources, and often large parts of the internet fit this bill. This includes online forums where, sometimes entirely unprovoked, bad language is prevalent. This language then ends up in the training data, and given that the LLM does not inherently understand the meaning behind this language, it will end up in the LLMs vocabulary, using this language against the users.

**Generative LLMs learn and perpetrate societal bias** Just as inappropriate language can end up in the LLMs vocabulary, biases might also be perpetrated. Biases are often prevalent in historical texts, e.g. suggesting women or men have rigid roles within society. These historical texts are still important to learn from, even if society has moved on from these believes. The LLM, unable to inherently discern between the past and current bias, might thus perpetrate past societal bias.

**What is intelligence?** Intelligence is defined in a multitude of ways. But all of these converge around the same core principles of this psychological element. Some argue that intelligence can be defined as the general mental ability for reasoning, problem solving and learning (Colom et al., 2010). Others would expand on this by adding planning, thinking abstractly, comprehending complex ideas, and differentiate between learning quickly and learning from experience (Gottfredson, 1997). A final addition to this definition is argued to be the inclusion of the ability to adapt effectively to the environment, and the ability to overcome obstacles by taking thought (Neisser, 1996).

# 3 METHODOLOGY

In this paper, we follow an explorative methodology by testing LLMs capabilities to play games. We start with current benchmark games in the field (e.g. chess) and, subsequently, we use the results to inform the next game choice. Each exploration aims to reveal certain aspects of the way LLMs interact with playful environments. The process is based on drawing hypotheses through speculation from the play results and testing the hypotheses in new appropriate game environments. All experiments are performed using the llama3.2:latest model (Meta, ).

## 3.1 Chess

The first game we employ to investigate the behavior of LLMs within games is chess. Chess is chosen as a starting point, aside from being the typical game on which AI has been tested historically, due to several properties that this game has. Firstly, chess is, compared to most modern video games, a relatively simple game: The entire game is played on an eight-by-eight board where both players have perfect information of all the pieces in play, meaning

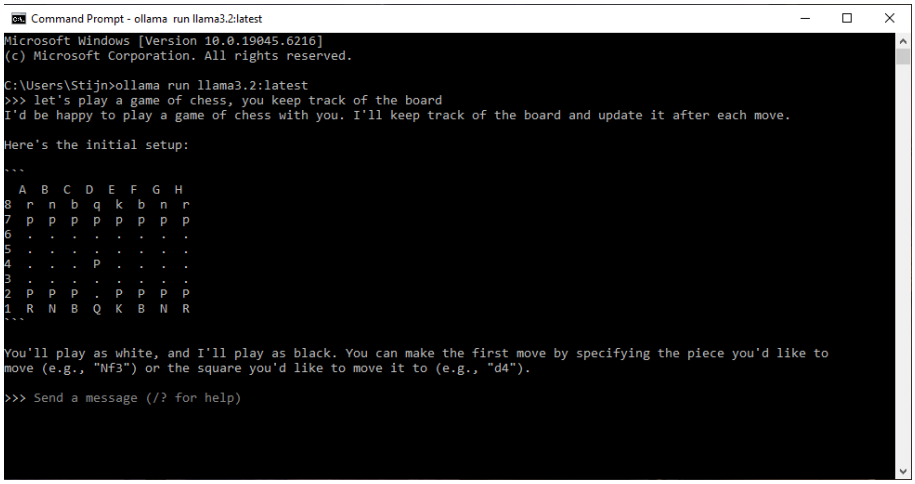


Fig. 3: Chess interface in the terminal. The LLM is prompted asking it to play chess, after which the LLM performs the first move. After each move the current state of the board is printed to the terminal.

that each player can see all pieces at all times. Furthermore, chess contains, while still vast, a finite number of possible legal positions ($10^{40}$ (Steinerberger, 2015)). This increases the likelihood of the LLM being able to develop strategies. Finally, the rules of chess are clearly defined. Seeing as chess is one of the most popular games ever created with a large backlog of strategies and tournament data, the LLM is highly likely to have been trained on the rules of chess and the strategies surrounding this game, making the LLM boast a larger understanding of chess than a more niche game which has not been represented in the training data. This allows us to simply prompt the LLM by asking it to play a game of chess, without being required to first explain the entire ruleset of chess, as can be seen in Figure 3. Our first attempt explored existing chess software, which was modified to directly take LLM output as input and convert this input into a legal move. Our other attempt explores playing chess against the LLM in the terminal, without intermediary software. Each player (LLM and prompter) takes turns performing a move. The LLM keeps track of the current state of the board and visualizes this in the terminal after every move.

### 3.2 Checkers

The second game we employ to investigate the behavior of LLMs within games is checkers. Checkers is structurally similar to chess in a multitude of factors: Both games are played on an eight-by-eight board, both players have perfect information of all the pieces, and there is a finite number of possible legal moves ($10^{20}$ assuming perfect play (Schaeffer et al., 2007)). The major difference between these games however, is that while chess uses multiple different kinds of pieces, each with their own set of possible legal moves, checkers only uses two kinds of pieces (a "man" and a "king"). This, combined with the fact that checkers is only played on the black squares of the board, makes the number of rules in the game of checkers a lot lower than chess. These factors reduce the action space which in turn reduces the complexity of the game. As a result, the LLM should have an easier time playing checkers than chess. Once again, we simply prompt the LLM, asking it to play a game of checkers. Both the LLM and the prompter take turns performing a move on the board, which is kept track of by the LLM. After every move, the LLM visualizes the board in the terminal.

### 3.3 Tic-Tac-Toe

The third game we employ to investigate the behavior of LLMs within games is Tic-Tac-Toe. Even simpler than both chess and checkers, Tic-Tac-Toe is played on a three-by-three board (grid). Once again, both players have perfect information and there are only a finite number of possible board states ($3^9$). On this board each player takes a turn placing their corresponding sign (cross or circle) until one player has three of their signs in a row, or the board is filled making no further move possible. Due to it's simplicity, the expectation is that an LLM should be able to successfully and competitively play a game of Tic-Tac-Toe. Once again, we simply prompt the LLM asking it to play a game of Tic-Tac-Toe, assuming

it already possesses knowledge of the rules. Both the LLM and the prompter take turns performing a move on the board, with the LLM keeping track of said board and visualizing the current state of the board in the terminal after every move that has been performed.

### 3.4 Zork

The final game we employ to investigate the behavior of LLMs within games is Zork. Rather than a game that takes place on a board with clearly defined rules, we opted for a game that should come more natural to an LLM. In particular, what makes LLMs so popular is their ability to process and reproduce text in a human-like fashion. Zork uses these conversational aspects by allowing the player to prompt the game with an action, to which the game replies with a scenario as a result of said action. This creates a back and forth between game and player not too dissimilar from a story being told. An example of such a scenario is shown in Figure 4.

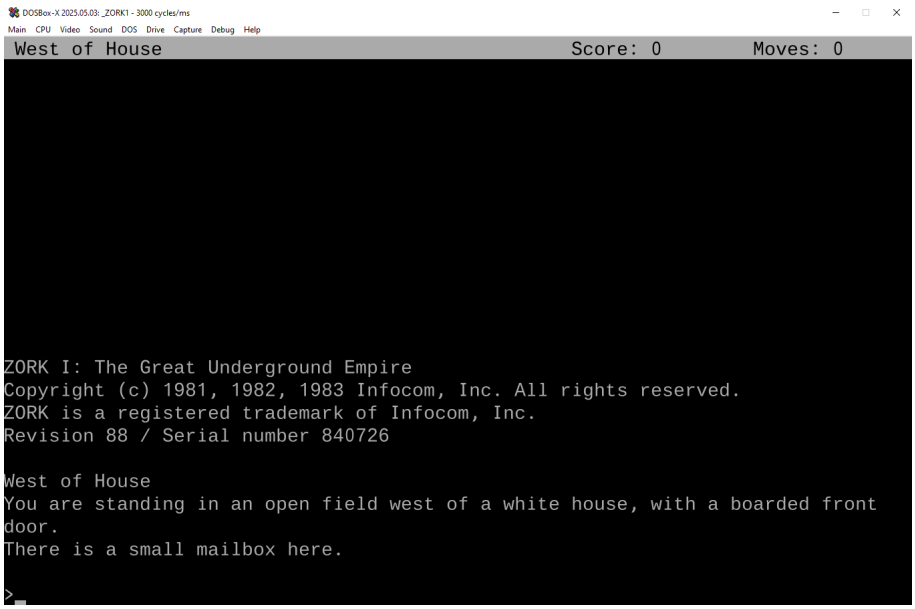


Fig. 4: The Zork interface. Shown is the initial scenario at the start of the game. The player can respond by providing an action and a subject of said action (e.g. open mailbox).

Zork requires input of a certain format, namely an action plus a subject which this action should be performed on. We've tested two different methods of playing Zork to see how good the LLM is at restricting itself to specific rules. The first method requires the LLM to provide its output in a specific format (action and subject of action) interpretable by Zork. The second method lifted this restriction, but still required the LLM to provide only one action per instruction. This allows the LLM to describe the thought process behind its action, as we served as an intermediary to reinterpret this output to an input acceptable by Zork. This reinterpreted output would then be funneled to Zork instead. In both cases the LLM would be prompted with the unfiltered output from Zork, having started the session with an explanation that it would be given a scenario and should reply with an action to perform in said scenario. The second method was found to be the most effective, and as such was chosen to play the actual game until we reached the cellar, accessible from the trap door in the white house. Every action (Zork output, LLM output and reinterpreted LLM output) is tracked in an excel sheet, which is analyzed for behavioral patterns the LLM exhibits during the playing of Zork.

## 4 RESULTS

### 4.1 Chess

Given that chess is a well-known game, simply prompting the LLM asking it to play a game of chess allowed us to immediately do this. We first used a separate chess program, which

automatically checked for the legality of moves performed, and fed it the LLM output in a format interpretable by the program. Legal moves would be performed, illegal moves would prompt the LLM letting it know it provided an illegal move and as such should retry. Both the prompter and the LLM took turns moving a piece on the chessboard. The prompter (user) would use their mouse to move pieces on a visualized chess board. These moves would then be translated to text and used as a prompt for the LLM. The LLM would reply in text with a desired move which was then translated to a move on this visualized chess board. Moves were done in the form *start_position* (e.g. e2) → *end_position* (e.g. e3). The built-in check for the legality of moves performed would often return that the LLM tried to perform an illegal move. Given that from the perspective of the prompter the entire process was similar to a black-box, we eventually opted for a terminal-based chess board, where each move was visualized by the LLM on a chess board it tracked itself. This was also done in the hope that having the LLM keep track of the board itself would give it a better understanding of the state of the board and thus positively affect its performance with respect to legal moves. Playing against the LLM, several problems became apparent:

- **The LLM often does illegal moves:** these moves include the LLM moving pieces that are blocked by their own pawns, to moving a piece in a wrong pattern (e.g. a knight that moves only forward).
- **The LLM moves the wrong pieces:** the LLM states it moves a piece from one location to another, but then moves a different piece altogether. Sometimes, the LLM moves using the opponents' pieces rather than its own color.
- **The LLM fails to reverse a move:** the prompter suggesting the last made move was illegal and should be reversed leads to the LLM either performing another move, without the reversal of the previous one, resulting in it performing two separate moves in one turn (of which at least one illegal) or it leads to the moving of wrong pieces, similar to the second issue.
- **The LLM does not understand strategies:** the LLM states it will perform a move according to a certain strategy (e.g. the Sicilian Defense) but then performs a completely different move.

### 4.2 Checkers

Similarly to chess, we simply prompted the LLM asking it to play a game of checkers. Given that both games are played on the same board (same coordinates), the LLM was able to present this board in the terminal. Each player took turns moving one of the pieces on the board by typing the original coordinate and the target coordinate. After multiple attempts to play checkers against the LLM, we observed the following patterns:

- **The LLM often does illegal moves:** these moves include the LLM moving pieces that are blocked by other pieces, to moving a piece in a wrong pattern (e.g. a piece moving onto another color).
- **The LLM moves the wrong pieces:** the LLM states it moves a piece from one location to another, but then moves a different piece altogether.
- **The LLM fails to reverse a move:** the prompter suggesting the last made move was illegal and should be reversed leads to the LLM either performing another move, without the reversal of the previous one, resulting in it performing two separate moves in one turn (of which at least one illegal) or it leads to the moving of wrong pieces, similar to the second issue.

### 4.3 Tic-Tac-Toe

Prompting the LLM asking it to play a game of Tic-Tac-Toe allowed us to do exactly that. The LLM kept track of the board and both prompter and LLM took turns performing a move. Each of the nine squares on the board was numbered allowing the prompter to perform a move by stating their sign (e.g. X) on a number corresponding to a square. Playing multiple games of Tic-Tac-Toe against the LLM revealed some issues:

- **The LLM fails to notice when the opponent has a setup that would make their next move a winning move:** when the prompter has a setup that would make their next move a winning move (e.g. two in a row with a third spot empty), the LLM fails to notice this and plays a move that does not block this winning move.
- **The LLM keeps playing even after a three in a row has been achieved:** when called out on this, the LLM does notice that a winning move has been made, but this suggests that it only does this when tasked to do so, rather than doing this on its own.

– **The LLM fails to reverse moves:** when tasked to reverse a move and try something else (e.g. when a winning move has been set up and the LLM performs a move that does not block this move), the LLM often fails to reverse this move. Similarly to chess, the LLM does not reverse the move at all, but instead either performs another move, or removes a wrong sign.
– **The LLM does not improve:** unless called out, it always performs the same moves, even if the result is always a loss. When suggested to try a different strategy, it still performs the same moves. Only when explicitly forbidden from performing those exact moves will it try something else, but will fall back into the same pattern the next game.

### 4.4 Zork

The final game we explored was Zork. First we prompted Zork directly with the output from the LLM, having instructed the LLM to only provide an action and a subject of this action. In turn the LLM was prompted directly with the output from Zork. This quickly proved to be difficult as the LLM kept suggesting actions or subjects foreign to Zork (e.g. "Search the house slowly" where Zork does not understand "slowly"). As such, we opted to work as an intermediary between the LLM and Zork. The LLM would still receive the output from Zork, but rather than adhering to a strict format usable by Zork, the LLM was now allowed to describe its intentions. This description would then be reformatted by the prompter and used as input for Zork. We ordered the problems we encountered during gameplay into three different degrees of severity depending on how much these problems affected the gameplay. The first category covers minor problems, e.g. logical fallacies, which did not immediately interfere with the gameplay. The second category contains moderate problems, e.g. the LLM losing its sense of directions, or when we had to interrupt and provide a suggestion because the LLM kept trying the same thing. These problems do affect gameplay to some degree and often require intervention. The final category covers the major problems, e.g. hallucinations by the LLM, which severely affect gameplay and require intervention, lest the LLM spirals in its own delusions. The problems, their number and severity have been visualized in Figure 5. We notice that the vast amount of problems fall under moderate severity which, combined

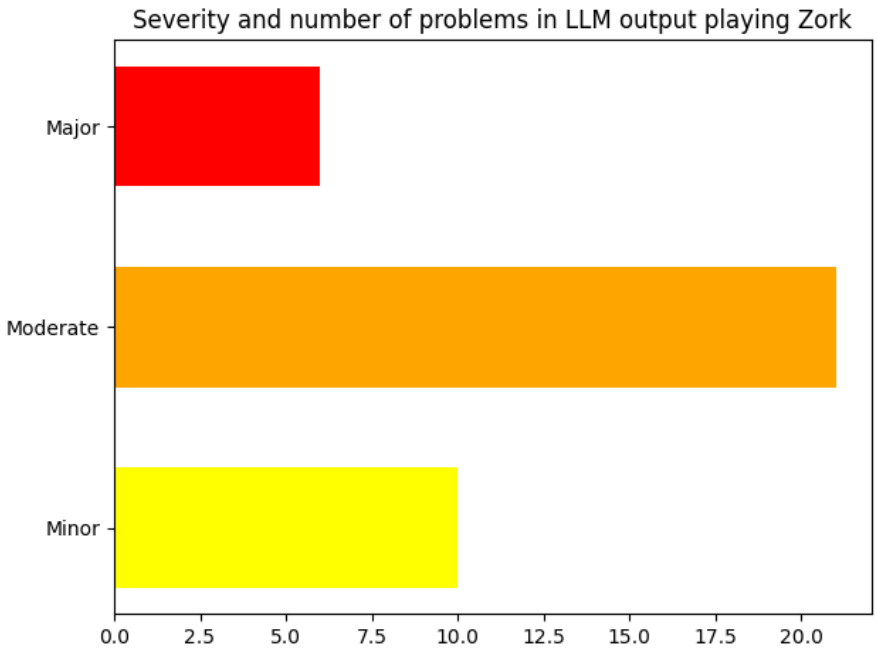


Fig. 5: Visualized is the count of problems encountered during the gameplay of the LLM in Zork. These problems have been categorized based on their severity, which indicates to what degree the problem affects gameplay and requires intervention. Out of the 151 total prompts to the LLM, 10 resulted in minor problems, 21 resulted in moderate problems, and 6 resulted in major problems.

with the problems making up the major severity category, means most (roughly 73%) of

the encountered problems require some degree of intervention. Further investigating the encountered problems revealed the following patterns:

- **The LLM hallucinates:** the LLM shows signs of hallucination, a common problem in LLMs. Examples include it suggesting to investigate a bush while Zork states there is none, or the LLM suggesting it is inside a time loop because it returned to a room it has been in before.
- **The LLM fails to try new things when stuck and often falls back into the same patterns:** similar to Tic-Tac-Toe, the LLM often falls back into the same patterns and fails to try new things when stuck. An example of this is when the LLM found itself in the house at the start of Zork. From the kitchen it was able to go upstairs, there was a passage that led east to the living room and a chimney that led down. In the kitchen there was a table with items and the living room contained a door, a carpet and a trophy case. After exploring these rooms and all the items they held, the LLM opted multiple times to return to the door and inspect it, expecting a change. This was done even while the attic (up the stairs) and the chimney were left unexplored. When suggested with exploring these places, the LLM moved from the living room to the kitchen, but immediately forgot why it moved to the kitchen, opting to move back to the living room and inspect the door again.
- **The LLM has a hard time with directions:** when it eventually reached the attic, signs that the LLM had a hard time with spatial awareness started showing. The living room was east of the kitchen, making the kitchen west of the living room. From the kitchen one can go up the stairs to the attic. When in the attic, the LLM suggested to move up the stairs to the living room or, when in the living room, the LLM suggested moving up the stairs to the attic, even though the stairs were in the kitchen.
- **The LLM has a hard time following instructions:** Zork takes one action and the subject of this action as input. As such, the LLM was tasked with providing only one action per prompt. This instruction was often ignored as the LLM often provided multiple actions (think a multi-step plan).
- **The LLM can keep track of inventory:** the LLM was able to keep track of the items it picked up along the way. A shortcoming in this is that the LLM figured that items it crossed along the way were automatically picked up. E.g. if the LLM came across a lantern, but did not explicitly pick this item up, it still figured it was in possession of this item.

# 5 DISCUSSION

In this chapter we analyze the results and report patterns found within these results. Furthermore, we compare encountered problems to known LLM problems and decide if the shown behavior can be considered intelligent.

## 5.1 Inherent comprehension of rules

Throughout the games, the LLM started to show several patterns. In chess (and similarly in checkers) we saw the LLM often doing illegal moves. These moves include bishops moving through a line of pawns, or pieces moving on the board in ways that they are not able to. Furthermore, the LLM states it will move according to a certain strategy, but then moves another piece altogether. A plausible cause for this is that the LLM, given the context of chess, "understands" that these strategies are associated with, and mentioned around the subject of chess. The LLM however does not understand these strategies and as such does not move according to these strategies. Similar problems arose in tic-tac-toe, where the prompter has a winning move set up, and the LLM failed to notice said winning move. One could argue that something as simple as checking if your opponent is about to win is a strategy. Furthermore, the LLM kept playing after the winning move had already been made. This further fuels the belief that the LLM does not play according to strategy (checking for winning moves) and simply produces board states that would be possible given the previous one. Tic-tac-toe also showed us that the LLM failed to improve: it would continuously perform the same moves, and when called out on this it would not change this "strategy" even though it always led to a loss. Understanding of the game, and with it the concepts of winning or losing, would lead any other player to try different strategies, as the current one always leads to failure. The LLM however, stuck to a failing strategy, as it did in Zork. In Zork the LLM would keep investigating the boarded front door hoping for a change. A player with basic understanding of the game, or games in general, would quickly realize that the door isn't going to spontaneously open and it might be more beneficial to explore yet unexplored

areas. The LLM however, gravitated to the front door, forfeiting progress in the process by seeing what other areas had to offer. These factors combined lead us to believe LLMs have an inherent lack of the comprehension of rules of games, and as a result are unable to device proper, or any, strategies in these games.

### 5.2 Understanding of spatial components

We've seen that several factors at which the LLM fails while playing games seem to be symptoms of a larger problem, the inherent lack of comprehension of the rules of games. However, it is likely that these factors are not simply symptoms of a single problem, but rather a multitude of problems. In chess and checkers we saw the LLM often performing illegal moves. We attributed this to the LLM not having a clear understanding of the rules of these games, thus not understanding how pieces are allowed to move. Another factor at play could be that the LLM does not have the "mental capabilities" to understand the spatial coordinates. For example, when we execute a move from e2 to e4, where 'e' is the column and numbers are the rows, humans can create a (mental) map of the board where a piece is moved two rows. The LLM, unable to create such a map, will not understand the meaning behind these coordinates and as such will be unable to move pieces in the correct fashion. This could also explain why the LLM used the wrong pieces altogether. If a move is made according to a coordinate that, according to the LLM, is a different coordinate altogether because it does not understand the coordinate, the wrong piece will be moved regardless of whether it is the piece of the prompter or the LLM. Further evidence of the LLM not understanding spatial components could be seen in Zork. The white house in Zork contained different rooms which could be reached by traversing in a certain direction from another room: the attic was up the stairs from the kitchen, the living room could be reached by taking a passageway east from the kitchen, and through the window west the player could traverse outside again. A simple mental map of these four spaces would give humans a clear sense of direction, yet the LLM showed that it had great difficulty with this: at points it figured the living room was upstairs of the attic, with the kitchen being east from the living room. This shows that not only does the LLM have a hard time with coordinates, but also direction. Together, this suggests that LLMs have difficulty understanding spatial components. Since coordinates and directions are relative to a point of origin, it is likely that this problem arises because of the inability of LLMs to create mental maps as humans are able to.

### 5.3 Understanding of temporal components

Not only have we seen the LLM show difficulty in understanding spatial components, we've also seen it struggle with temporal data. In chess and checkers the LLM would often perform illegal moves, which we would attribute to its lack of understanding of spatial components. To continue the game, these moves should have been corrected so that a legal move could be executed, thus progressing the game. While the reversal of the previous move should be as simple as fetching the previous board state, after which a legal move could be executed, the LLM failed to do so. Rather, the LLM performed another move on the most recent board, suggesting that it has no concept of a "most recent move" given the game of chess. The arguably simpler game of checkers showed the same signs: an illegal move was not reversed by the LLM when tasked with doing so, but rather another move was performed on the current board. In the even simpler game of tic-tac-toe, while not necessarily performing illegal moves, the LLM still failed to reverse moves when tasked with doing so. These factors combined suggest that it might not be a problem with the difficulty of the game, but rather a problem with how the LLM processes temporal data. Another factor that plays into this is how the LLM processes time in Zork. When the LLM got stuck, it would wait, hoping that something would change in the environment. When this was not the case, it would draw the conclusion that it had gotten itself into a time-loop. The same conclusion would be drawn when the LLM would return to a room it had just been in, only to find everything the same. While less evident than the previous signs, these could still signal that the LLM has a hard time processing data related to time, possibly due to a lack of understanding of the meaning of time.

### 5.4 Following instructions

We've seen the LLM fail at several factors present within the games. However, the LLM showed great difficulty in handling things external to these games. We saw the LLM struggling in chess with outputting the coordinates in a specific format interpretable by the chess

client, when it was instructed to do so. Since it failed to consistently do so, we were forced to play chess against the LLM in the terminal, having the LLM keep track of the board instead. Similar problems arose in Zork where the LLM was instructed to provide instructions of a format that would be interpretable by Zork. Failure to do so led to us reinterpreting the LLM instructions instead. However, the LLM was still instructed to only provide one action at a time yet still failed in adhering to this rule, providing multiple actions in the same output. In Zork we also saw the LLM convinced of it being in a time loop. When explicitly told that this was not the case, the LLM would let it be but eventually would fall back to its old habits and propose the idea of a time loop again, essentially ignoring the earlier instructions. These factors combined suggest that the LLM not only struggles with understanding the games, but also struggles in playing these games within the confines of the provided instructions. This could suggest that the LLM simply does not understand the instructions just as it does not understand the games, and is thus unable to follow said instructions.

### 5.5 LLM intelligence

We've observed the behavior of the LLM within the games, and have classified each of these behaviors within one of four categories. Earlier we have highlighted previous research that noted multiple known shortcomings in LLMs. Many of the problems encased within our categories can be tied to one or multiple of these shortcomings:

One of these shortcomings is that LLMs are only aware of what they saw during training. An example of this is that during training the names of strategies in chess might have been presented to the LLM, but the actual board states and meaning behind these strategies were not presented to the LLM, at least not to the degree that the names of the strategies were. As such they simply mention a strategy, "knowing" it is related to chess, but have no understanding of what this strategy actually constitutes. This same argument can be extended to Zork, where in the training set the LLM was taught that a door is something associated with progression, as it usually leads to a new room. As a result the LLM would gravitate to the door rather than the stairs, which might have been represented less in the training set. We've also seen the LLM show signs of fast forgetting or a short attention span. This was most evident in its lack of following instructions, where it would either follow the instruction it was provided for a few iterations, after which it would ignore said instruction again, or it would simply ignore the instruction from the beginning. Given that we used an LLM with a smaller size, the context window is also smaller as a result, which could lead to the LLM forgetting about prior instructions. We speculate that this is also part of the reason why the LLM was unable to retrieve prior board states, when asked to reverse a move.

Yet another, and maybe the most prominent shortcoming we've seen, is the LLM showing signs of trouble with reasoning. Given that LLMs have been trained to generate the most probable response to a prompt, their reasoning abilities do not exceed that which they've seen in their training. This allows it to play the simple game of tic-tac-toe (with still some shortcomings present), but the harder and more complex games of chess and checkers far exceed its reasoning abilities. It performs exceedingly well in a game like Zork, which is arguably more complex than tic-tac-toe, but the medium of a conversational based game is more familiar to the LLM, increasing its familiarity with the context and with it the reasoning capabilities of the LLM. This would suggest that the LLMs reasoning abilities are context dependent.

**Intelligence** We've highlighted related work which defines intelligence. To answer whether the behavior of the LLM within these games can be considered intelligent, we refer to this definition of intelligence. Although the LLM did exhibit factors such as basic reasoning and basic problem solving, the vast majority of factors were not present in the LLM. We assume this led to the problems the LLM displayed in the playing of the games. This would suggest that the LLM, based on the earlier defined definition of intelligence, does not exhibit intelligent behavior within these games, which in turn could suggest that LLMs are not intelligent.

## 6 CONCLUSION

We used games as testbeds to observe LLM behavior. To this end we tested an LLMs capabilities to play in several games. We started with chess, the traditional benchmark in the

field, and let the choice of subsequent games be informed by the results of the prior. The LLM was shown to have difficulty in chess, often performing illegal moves. We attributed this to a lack of reasoning in the LLM, requiring a game with lower complexity and with it a lower requirement of reasoning abilities. Due to it being played on the same board as chess yet having a lower complexity, checkers was chosen as the next game to be played. However, it was quickly revealed that checker posed the same problems for the LLM as chess had prior, and as such the need for an even simpler game was revealed. Remaining within the realm of board games, we opted for tic-tac-toe, arguably one of the simplest games. While not performing moves as problematic as in chess and checkers, the performance of the LLM in tic-tac-toe still showcased a lack of reasoning abilities. This suggested that board games might be too complex for LLMs altogether, leading us to play Zork, a text-based game which should come more natural to LLMs. While the LLM, by being able to progress, did perform better within Zork than the board games, its shortcomings were still evident. The LLM often got stuck repeating the same patterns and showed a lack of spatial understanding.

Analyzing the behavior the LLM showcased within these games and comparing it to known problems with LLMs, as well as the definition of intelligence as defined by experts, leads us to believe that the behavior displayed by the LLM can not be considered intelligent.

### 6.1 Limitations and Future Research

It should be noted that our choice of games was subjective. Based on how we interpreted the results from each game, the choice of subsequent games was made. Meaning a different interpretation of the results, or different observations altogether, might lead to different conclusions and as such a different choices of subsequent games. Future research could explore the same method we employed and investigate if either the same behavior is exhibited and the same conclusions are drawn, or if different behavior can be noted leading to possibly different conclusions.

Future research could investigate whether the same behavior is showcased within other LLM models. The current observations are made based on one specific model which, when compared to the known problems of LLMs, does exhibit the expected behavior. It should still be worthwhile to compare these results to those that would be obtained from other models, as this could further solidify and even expand the drawn conclusion. These models could include both larger models, as well as different models (architectures) altogether.

Finally, future research could build upon our method by employing more games based on the obtained results, as our current method explored only four games. Continuing this procedure should result in clearer analyzable patterns, which could serve as a predictor to which games LLMs might or might not be able to succeed in.

**Acknowledgments.** The author would like to thank Giulio Barbero and Mike Preuss for their guidance during the project, and his parents for their continued support.

Alhafni, B., Vajjala, S., Bannò, S., Maurya, K. K., and Kochmar, E. (2024). Llms in education: Novel perspectives, challenges, and opportunities.

Arulkumaran, K., Cully, A., and Togelius, J. (2019). Alphastar: an evolutionary computation perspective. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, page 314–315. ACM.

Babu, M. A., Yusuf, K. M., Eni, L. N., Jaman, S. M. S., and Sharmin, M. R. (2024). Chatgpt and generation 'z': A study on the usage rates of chatgpt. *Social Sciences Humanities Open*, 10:101163.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2020). Emergent tool use from multi-agent autocurricula.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F., and Lin, H.-T., editors, *Advances in Neural Information Processing Systems (NeurIPS 2020)*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bureau, E. U. N. (2025). Survey: 52% of u.s. adults now use ai large language models like chatgpt.

Campbell, M., Hoane, A., and hsiung Hsu, F. (2002). Deep blue. *Artificial Intelligence*, 134(1):57–83.

Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T. B., Song, D., Úlfar Erlingsson, Oprea, A., and Raffel, C. (2021). Extracting training data from large language models. In Bailey, M. and Greenstadt, R., editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2633–2650. USENIX Association.

Churchland, P. M. and Churchland, P. S. (1990). Could a machine think? *Scientific American*, 262(1):32–39.

Colom, R., Karama, S., Jung, R. E., and Haier, R. J. (2010). Human intelligence and brain networks. *Dialogues in Clinical Neuroscience*, 12(4):489–501. PMID: 21319494.

Deane, P. (1979). *The first industrial revolution*. Cambridge University Press.

Deterding, S., Dixon, D., Khaled, R., and Nacke, L. (2011). From game design elements to gamefulness: defining" gamification". In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*, pages 9–15.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Ettinghausen, J. (2025). 18 months. 12,000 questions. a whole lot of anxiety. what i learned from reading students' chatgpt logs.

Farhi, F., Jeljeli, R., Aburezeq, I., Dweikat, F. F., Al-shami, S. A., and Slamene, R. (2023). Analyzing the students' views, concerns, and perceived ethics about chat gpt usage. *Computers and Education: Artificial Intelligence*, 5:100180.

Ferreira, M., Pinha, A., Fonseca, M., and Lopes, P. (2023). Behind the door: Exploring horror vr game interaction and its influence on anxiety. In *Proceedings of the 18th International Conference on the Foundations of Digital Games*, pages 1–11.

Glaese, A., McAleese, N., Trebacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., Campbell-Gillingham, L., Uesato, J., Huang, P.-S., Comanescu, R., Yang, F., See, A., Dathathri, S., Greig, R., Chen, C., Fritz, D., Elias, J. S., Green, R., Mokrá, S., Fernando, N., Wu, B., Foley, R., Young, S., Gabriel, I., Isaac, W., Mellor, J., Hassabis, D., Kavukcuoglu, K., Hendricks, L. A., and Irving, G. (2022). Improving alignment of dialogue agents via targeted human judgements.

Gómez-Maureira, M. A., van Duijn, M., Rieffe, C., and Plaat, A. (2022). Academic games - mapping the use of video games in research contexts. In *Proceedings of the 17th International Conference on the Foundations of Digital Games*, FDG '22, New York, NY, USA. Association for Computing Machinery.

Gottfredson, L. S. (1997). Mainstream science on intelligence: An editorial with 52 signatories, history, and bibliography. *Intelligence*, 24(1):13–23. Special Issue Intelligence and Social Policy.

Haltaufderheide, J. and Ranisch, R. (2024). The ethics of chatgpt in medicine and healthcare: a systematic review on large language models (llms). *npj Digital Medicine*, 7(1):183.

Hoekenga, B. C. (2007). *Mind over machine: what Deep Blue taught us about chess, artificial intelligence, and the human spirit*. PhD thesis, Massachusetts Institute of Technology.

Iqbal, A. (2010). The relevance of universal metrics in relation to human aesthetic perception. In *2010 International Conference on Intelligent and Advanced Systems*, pages 1–6.

Jones, C. R. and Bergen, B. K. (2025). Large language models pass the turing test.

Klopfer, E., Reich, J., Abelson, H., and Breazeal, C. (2024). Generative AI and K-12 Education: An MIT Perspective. *An MIT Exploration of Generative AI*. https://mit-genai.pubpub.org/pub/4k9msp17.

Kucharavy, A. (2024). Fundamental limitations of generative llms. In *Large Language Models in Cybersecurity: Threats, Exposure and Mitigation*, pages 55–64. Springer Nature Switzerland Cham.

Liao, Y., Loures, E. R., Deschamps, F., Brezinski, G., and Venâncio, A. (2018). The impact of the fourth industrial revolution: a cross-country/region comparison. *Production*, 28:e20180061.

Meta. Llama3.2.

Milmo, D. (2023). Chatgpt reaches 100 million users two months after launch.

Miyazaki, K. and Sato, R. (2018). Analyses of the technological accumulation over the 2nd and the 3rd ai boom and the issues related to ai adoption by firms. In *2018 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1–7.

Mokyr, J. and Strotz, R. H. (1998). The second industrial revolution, 1870-1914. *Storia dell'economia Mondiale*, 21945(1):219–245.

Neisser, U., B. G. B. T. J. J. B. A. W. B. N. C. S. J. H. D. F. L. J. C. P. R. S. R. J. U. S. (1996). Intelligence: Knowns and unknowns. In *American Psychologist*, page 77–101.

Nyandwi, J. (2023). Ai research blog - the transformer blueprint: A holistic guide to the transformer neural network architecture.

Powell, G., Lillrank, L., Lillrank, I., and Lee, A.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Roberts, B. H. (2015). The third industrial revolution: Implications for planning cities and regions. *Work. Pap. Urban Front*, 1(1):1–21.

Schaeffer, J., Burch, N., Björnsson, Y., Kishimoto, A., Müller, M., Lake, R., Lu, P., and Sutphen, S. (2007). Checkers is solved. *Science*, 317(5844):1518–1522.

Shaikh, T. A., Rasool, T., Veningston, K., and Yaseen, S. M. (2025). The role of large language models in agriculture: harvesting the future with llm intelligence. *Progress in Artificial Intelligence*, 14(2):117–164.

Shi, D. (2024). Use chatgpt to maximize everyday efficiency. *Geology and Geochemistry of the Only Independent Tellurium Deposit in the World*, 1(1):87–99.

Shuster, K., Komeili, M., Adolphs, L., Roller, S., Szlam, A., and Weston, J. (2022). Language models that seek for knowledge: Modular search  generation for dialogue and prompt completion.

Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm.

Steinerberger, S. (2015). On the number of positions in chess without promotion. *International Journal of Game Theory*, 44(3):761–767.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention is all you need.

Yenduri, G., M, R., G, C. S., Y, S., Srivastava, G., Maddikunta, P. K. R., G, D. R., Jhaveri, R. H., B, P., Wang, W., Vasilakos, A. V., and Gadekallu, T. R. (2023). Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions.