Sensitivity of Different LLMs to Prompt Representations in AI Planning Problems

JIMMY GAN*, Australian National University, Australia
PHILIP CAISIP*, Australian National University, Australia
RAHUL SHOME, Australian National University, School of Computing, Australia
HANNA KURNIAWATI, Australian National University, School of Computing, Australia
c

As AI planning problems become more complicated, many are looking at using Large Language Models (LLMs) as a solution. The recent release of new LLMs presents an opportunity to see how they can help in planning. We investigate the capabilities of state-of-the-art LLMs in the domain of AI task planning. We explore how LLM performance varies with problem complexity, determined by block count. We also investigate how input prompt structure — whether prompts are given in natural language or planning domain definition language (PDDL) — affects the quality of generated plans. Our findings reveal that even the latest LLMs struggle to reliably reason about tasks and create sound plans independently. While GPT4 models show comparatively better performance, none excel in plan generation, raising questions about their suitability for precision-demanding applications, especially in complex scenarios. We observe a performance decline with increased problem complexity and identify issues with redundant steps in generated plans. Our study highlights the need for future research to improve the LLMs' planning efficiency, consider data quality, and explore the nondeterministic nature of LLM outputs.

Additional Key Words and Phrases: AI Planning, Task Planning, LLM, Large Language Models, GPT, Bard, LLaMA2, PaLM, Blocksworld

ACM Reference Format:
Jimmy Gan, Philip Caisip, Rahul Shome, and Hanna Kurniawati. 2023. Sensitivity of Different LLMs to Prompt Representations in AI Planning Problems. *Proc. ACM Meas. Anal. Comput. Syst.* xx, y, Article 777 (September 2023), 13 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Large Language Model (LLM) technology continues to advance at a tremendous rate. Waves of new models have been published in the last several years, each with their own unique features and claims. As a result, LLMs continue to be treated as state-of-the-art in the general field of natural language processing (NLP). Alongside OpenAI's work, models such as PaLM [4], LLaMA 2 [23], Gopher [15], Megatron-Turing NLG [18], and Chinchilla [7] have been developed to regularly push the limits of NLP. Considering that the first release of Vaswani et. al.'s

*Both authors contributed equally to this research.

Authors' addresses: Jimmy Gan, Australian National University, The Australian National University Canberra ACT 2600 Australia, Canberra, ACT, Australia, 2600, u7461674@anu.edu.au; Philip Caisip, Australian National University, The Australian National University Canberra ACT 2600 Australia, Canberra, ACT, Australia, 2600, u6971079@anu.edu.au; Rahul Shome, Australian National University, School of Computing, The Australian National University Canberra ACT 2600 Australia, Canberra, ACT, Australia, 2600, rahul.shome@anu.edu.au; Hanna Kurniawati, Australian National University, School of Computing, The Australian National University Canberra ACT 2600 Australia, Canberra, ACT, Australia, 2600, hanna.kurniawati@anu.edu.au.

2476-1249/2023/9-ART777 $15.00
https://doi.org/XXXXXXX.XXXXXXX

paper detailing transformers was in 2017 [26] and OpenAI's start with GPTx series was in 2018 [14], LLMs have come a long way in a short amount of time.

Among the various claims others have made about what LLMs can do is the ability to synthesise and reason about plans — that is, sequences of steps to take to achieve some goal. Considering that LLM architectures are fundamentally designed to complete text given some natural language prompt, the idea of LLMs developing an emergent capability to reason is seen as both interesting and scary. Many papers showcase LLMs performing things like explaining their responses to questions step-by-step [9, 27, 29], explaining jokes [21], and specifying actions for another AI agent to fulfill a human request [2, 32, 33]. An easy claim to make from all of this is that LLMs have some sort of hidden understanding about their prompts and are somehow internally "thinking through" their answer to provide a coherent final response, as humans do. Alternatively, there are efforts made by other papers to convey the opposite, that LLMs are not capable of reasoning or planning at all. Two examples of this are [24] and [16]. The primary implication that these types of papers have is that any successful results that LLMs may have with reasoning or planning is not because of actual reasoning but something else, likely the ability to copy from closely-related arguments sourced from its vast training corpus.

In this paper, we investigate the capabilities of new, state-of-the-art LLMs by themselves in AI task planning. We are interested in their abilities to draw conclusions from actions and the changes caused by the actions, which are necessary for planning, and how they compare against one another. To gain insight on this ability, we ran tests that involved passing problem specifications of a frequently used AI planning domain BlocksWorld to the various LLMs and prompting them to list the sequence of steps needed to be taken to reach a goal state. We also explore how these abilities differ as the problem domain becomes more complicated, determined by how many blocks are in the problem and how many steps there are in the optimal solution. Moreover, we investigate whether changing the structure of the input prompt changes any results. Namely, our tests includes cases that have a natural language prompt and cases that use planning domain definition language (PDDL) [11].

The main motivation of this study is to uncover where LLMs can be useful in AI planning and fields that use AI systems to plan, such as robotics. As LLMs are already seeing regular use in applications that resemble AI planning, it is useful to determine whether efforts are misplaced with LLMs or whether there is untapped potential still to be discovered. Indeed, LLMs attract a lot of attention from researchers and the general public alike. Their availability and ability to interpret natural language instructions make them an attractive choice for planners, especially for those that interact regularly human beings. However, better decisions regarding their use can be made with empirical evidence of their performance.

Our primary contribution is to show that, even now, state-of-the-art LLMs still cannot reliably reason about tasks and create sound plans by themselves. Tests run with PaLM, LLaMA 2 (70B parameters), Bard [13], and different versions of GPT3.5 [3] and GPT4 [12] all show low success rates for plans to solve BlocksWorld problems. This result applies to all problem complexities and input structures tested.

Section 2 details the relevant previous work in this field, including similar experimental investigations using LLMs as planners. Section 3 outlines our experimental setup, as well as gives an overview on necessary background — namely LMMs, PDDL, and BlocksWorld. Section 4 provides our results. Section 5 discusses the implications of the main findings as well as additional observations. Section 6 discusses future work.

2 RELATED WORK

Studies investigating how well LLMs can plan on their own have been conducted before, with the most notable work done in [24, 25]. This research provides a whole test suite and framework to investigate LLM planning capabilities, similar to our work. Also like ours, their test cases include problems that range in complexity, from simple problems that need one or two moves to problems that require long-horizon plans and a more sophisticated way of dealing with the effects of actions, such as understanding state changes. However, their investigations are limited by the LLMs that they test with. They test on GPT3 (davinci), Instruct-GPT3 (test-davinci-002) and BLOOM (170B parameters), which would have been state-of-the-art at the time of their project's start. Since LLMs have been one of the fastest advancing technologies, it is important to investigate further on the planning capabilities of LLMs as new models are made available. Furthermore, [24, 25] focuses on using few-shot prompting. There are currently no studies on new state-of-the-art LLMs that perform zero-shot tests, meaning prompts are not prefaced with examples of other prompts and their correct completions. Their conclusions are that GPT3, Instruct-GPT3, and BLOOM by themselves performed dismally in all of their test categories, including plan generation, generalisation, and reuse.

A similar study was conducted by [16] in parallel to the work done in [24, 25]. The tests done in this study primarily used PDDL to convey the problem specifications in the LLM prompts, also in a few-shot fashion. Compared to [24, 25], the researchers take a different approach to testing, with the most notable differences being their larger collection of different domains and their varying approaches to input generation. For example, in one set of tests, they replace some of the vocabulary used in the problem specification with a random commonly-used English word. They do so to determine if the quality of plans that their chosen LLM generates change depending on domain or whether the actions have semantic meaning as a natural language statement. Additionally, their study was done only using one LLM, a version of GPT3 called Codex. Similar to [24, 25], they conclude that Codex by itself largely fails as an AI planner.

Before these works, several other experiments were run on various LLMs, serving as pilot investigations on LLM reasoning capabilities. However, many of these tests were limited to relatively simple test cases. GPT3 was shown to "reason" about different topics like social scenarios and biological phenomena [10]. However, the result of their tests showed mixed outcomes. Others have taken a negative stance on LLM reasoning, with tests on Gopher performed by [15] showcasing subpar results. Contrasting this, several papers have been written on LLMs performing well in problem areas such as arithmetic and commonsense reasoning [9, 27, 29]. These three papers all employ a prompt-engineering technique named chain-of-thought prompting, which involves invoking the LLM to include in its final response a sequential step-by-step argument that explains why it gave that answer. Increased performance is shown in [29] when chain-of-thought prompting is used with few-shot prompting, while [9] sees similar improvements with zero-shot. It is tempting to infer that using chain-of-thought prompting forces the LLM to "think" through its answer before outputting it. Yet, this completely goes against the previously mentioned papers arguing that LLMs can do no such thing. It is important to keep in mind that these two sets of papers tend to use different problem domains for tests, which likely contribute to their positive or negative results. Domains such as arithmetic and commonsense often have similarly structured chain-of-thought explanations and solutions available on web resources, which are commonly used as part of LLM corpuses. In contrast, PDDL solutions are less commonly found.

In parallel to these studies, numerous systems have been developed, robotics or otherwise, each with their own way of utilising LLMs for planning-like purposes. A common tactic is to use LLMs in tandem with other systems, including other LLMs, to get better performance or guarantees in results. As demonstrated by [24], within the BlocksWorld testing suite, when GPT3 and BLOOM are combined with another autonomous planner LPG [5], the quality of the final plans generated improve dramatically. Other papers are finding uses for LLMs with various servant-like robots in kitchen and dining settings. LLMs can be seen performing many functions, such as translators to convert natural language commands into a more AI-friendly structure [22], leveraged for heuristic-based decision-making systems [2, 19, 31], or generators of rough plans that can be iterated on or refined by other systems [25]. Outside of robotics, LLMs are being used to plan for other AI agents. A surprisingly popular application is the use of LLMs to generate actions for an AI agent in a video game, with "Minecraft" [20] being a prominent example [8, 28, 33]. In order to get Minecraft achievements, many of these systems prompt LLMs for the correct sequence of tasks to do and actions to take, an application that is at least akin to "plan generation".

3 METHODOLOGY

Here, we give an overview on how we conducted the experiments. Our tests consisted of 50 Blocksworld problems in total, ranging from 3-block to 7-block problems with each number featuring 10 test cases. The problems were converted into appropriate prompts that were used to query each of the LLMs, using their respective APIs, and a plan was formed from their response. Our tests were zero-shot, meaning no example problems and solutions are included in the LLM prompt — only the relevant problem to solve. The subsections below detail how we handled each process.

3.1 Background

Large Language Models, based on the transformer architecture [26], fundamentally are models created to predict the next word given a sequence of words, often called a prompt, as input. By doing continuous sequential predictions or auto-completions, they are able to form coherent sentences in natural language. They are characterized by their vast size, encompassing billions of parameters and using a massive corpus likely sourced from the web as training data. This enables them to process and generate text with human-like fluency and coherence. They are also often fine-tuned to auto-complete in a way that makes them sound like a "chatbot" engaging in a conversation with the prompter. The outputs given by these "chatbot" LLMs are often called responses.

Planning Domain Definition Language (PDDL) [11] is a formal and expressive language widely employed in the field of artificial intelligence and automated planning. Developed as a standardized notation, PDDL serves as a means to rigorously specify the structural and procedural aspects of planning problems, enabling precise problem formulation. In short, it precisely specifies the initial state, valid actions allowed in a plan, and goal state of a given problem.

BlocksWorld is a planning domain commonly used in artificial intelligence and robotics, serving as a test-bed to evaluate aspects of planning, reasoning, and manipulation. It has seen use in research as early as 1972 [30] and is listed as an NP-hard problem set, as stated in [6]. In BlocksWorld, the objective is to rearrange a set of small blocks on a tabletop to achieve a predefined configuration, adhering to a set of rules that constrain block movement. Its abstraction and universality make it suitable for exploring fundamental questions in automated reasoning and robotics, making it the ideal choice for this research.

3.2 Tested LLMs

Our tests were run on nine LLMs in total. These are GPT3.5-turbo, GPT3.5-turbo-0301, GPT3.5-turbo-0613, GPT4, GPT4-0314, GPT4-0613, PaLM, Bard, and LLaMA2 (70B chat). These were accessed through their organisation-specific APIs, with the exception of LLaMA2 which was accessed through the Replicate API [1]. This means that not all LLMs were run in the same environments. The tested LLMs and their respective parameter counts are listed in Table 1 in order of release. The parameters counts also give insight on how open each of the organisations are with their technology.

3.3 Prompt Structure

Problems in the Blockworld domain are generated using tools provided by [17]. These problems are then converted into three forms, resulting in a test suite of 150 tests for each LLM. These forms are natural language, PDDL, and a "mystery" domain specification also given in natural language. An example of the three prompts specifying the same problem is given in Table 2.

The mystery domain, inspired by [25], rewords the BlocksWorld domain to include animals that are either inside or outside of other animals, instead of blocks being on top of other blocks. Functionally, this behaves the same as the standard BlocksWorld domain. This was added to determine whether LLM plans are robust

LLM	**Parameters (B)**	**Release Date**
GPT3.5-0314	N/A	14 Mar 2023
GPT4-0314	N/A	14 Mar 2023
Bard	137	21 Mar 2023
PaLM	540	10 May 2023
GPT3.5-0613	N/A	13 Jun 2023
GPT4-0613	N/A	13 Jun 2023
LLaMA2	70	18 Jul 2023
GPT3.5	N/A	22 Aug 2023
GPT4	N/A	22 Aug 2023

Table 1. Tested LLMs and their number of parameters in billions. LLMs are listed in release order. N/A parameter values indicate that the LLMs parameter count is not disclosed officially

Natural Language	Mystery	PDDL
There are 3 blocks. Block 1 is on top of Block 3, Block 2 is on top of Block 1, Block 3 is on the table. The goal is to move Block 1 on top of Block 2, Block 2 on the table, Block 3 on the table, how to achieve this? You may move a block only if no block is on top of it, and move a block on top another block only if no block is on top of both blocks	There are 3 Animals. tiger is inside duck, chicken is inside tiger, duck is outside. The goal is to move tiger inside chicken, chicken outside, duck outside, How to achieve this? You may move a Animal outside only if no Animal is inside it, and move a source Animal inside another target Animal only if no Animal is inside the source and target Animal.	The following PDDL describes a problem in the blocksworld domain: `(define (problem BW-3-42-1) (:domain blocksworld) (:objects b1 b2 b3) (:init (on b1 b3) (on b2 b1) (on-table b3) (clear b2) ) (:goal (and (on b1 b2) (on-table b2) (on-table b3) ) ) )` You may move a block only if no block is on top of it, and move a block on top another block only if no block is on top of both blocks

Table 2. Example problem given in three different input structures

against changes in the semantic meaning of actions, which can give extra insight as to whether they are logically reasoning about their plans or not.

3.4 Response Verification

The raw responses from the LLMs came in different structures themselves, often with "flavour-text" that did not contribute to the solution. In order to verify the validity of the plans, these responses needed to be converted to a standard format. For similarly structured responses, usually sourced from the same LLM, a template-based translator program was used. Otherwise, the translation was done by hand.

With the plan in a standard format, another verifier program took the plan as input and determined whether the plans achieved the goal state, did not achieve the goal state, or contained invalid actions. =-0wq It should be noted that many of the responses contained steps that are not explicitly part of the formal domain but behaved functionally the same as other steps. For example, some plans involved placing blocks on the floor or placing blocks on top of one of many tables. Even though these were not explicitly stated in the problem specification, these types of steps essentially achieve the same thing as placing a block on the sole table in the formal domain. We chose to employ a "softer" validation strategy and converted these steps into their formally-valid counterparts.

4 RESULTS

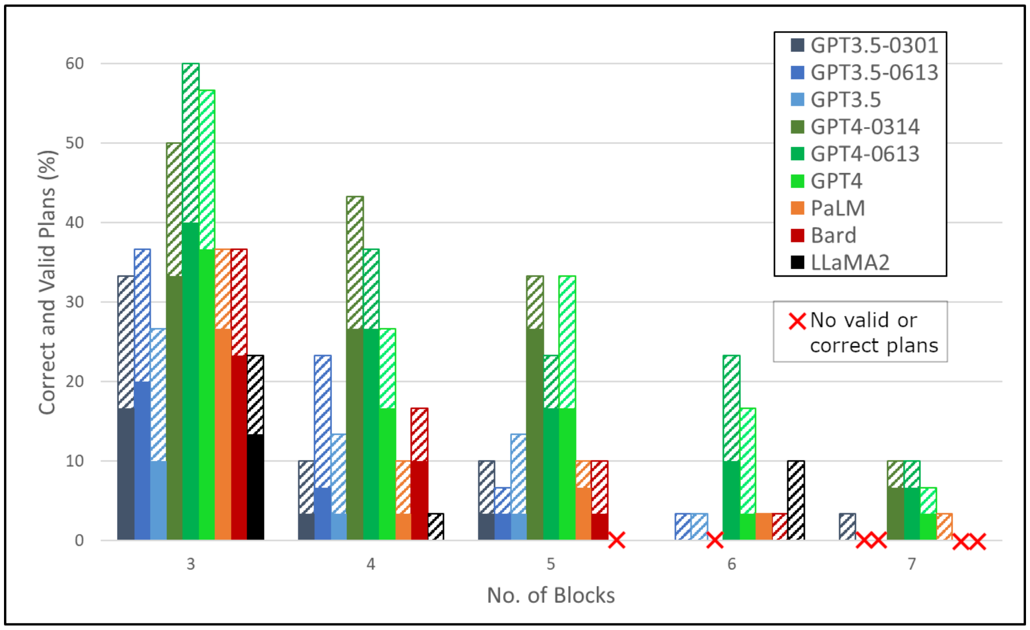

Fig. 1. Proportion across all three prompts structures of correct (solid bars) and valid (striped bars) plans generated by each LLM, broken down by number of blocks in the problems. GPT3.5 and GPT4 represent the latest builds of the respective LLMs as of September 2023.

Figures 1 and 2 each display the total proportion of correct and valid plans generated across the 150 test cases. A valid plan is defined to be a plan with no invalid moves and a correct plan is a plan that successfully ends with the goal state. The raw proportions of correct plans is given in Table 3 as an appendix.

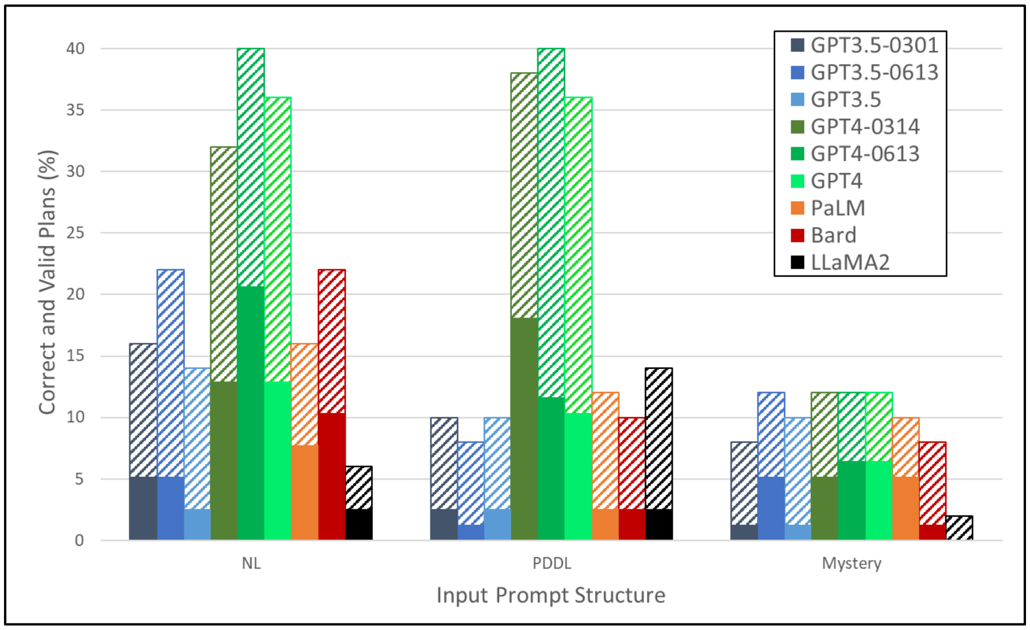

Fig. 2. Proportion across all 50 problems of correct (solid bars) and valid (striped bars) plans generated by each LLM, broken down by prompt structure. GPT3.5 and GPT4 represent the latest builds of the respective LLMs as of September 2023.

GPT3.5-turbo exhibited a modest level of performance, with 10% of its plans being correct and 25% being valid. Its performance deteriorated as the complexity of the questions increased, experiencing a pronounced decline in correct plans at 4-block tests and eventually reaching 0 correct plans for 6 and 7-block tests. Notably, an earlier iteration, GPT3.5-turbo-0301, displayed marginally superior performance, generating correct plans 16% of the time and valid ones 33% of the time for 3-block tests. In contrast, GPT3.5-turbo-0613, another earlier build of GPT3.5-turbo, emerged as the most proficient model among the GPT3.5 family, achieving a 20% rate of correctness for 3-block tests. Both of these models, however, follow the same pattern of diminishing correctness, ultimately reaching 0% for 5-block tests and beyond. In fact, all LLMs except for the GPT4 variants fail to produce any correct plans for problems with more than 5 blocks.

The GPT4 variants, in contrast to its predecessors, demonstrate a substantial improvement in tackling short-horizon problems, boasting 30%+ rates of correctness for 3-block tests — almost doubling the performance of the GPT3.5 variants. They consistently produce the most valid and correct plans out of all of the tested LLMs, even for long-horizon problems. Nevertheless, their efficacy still plummets significantly as the question became more intricate, with the proportion of correct plans diminishing to around 20% for 5-block problems and less than 10% for 6 and 7. Oddly, the newest version of GPT4 fails to generate any valid plans for 6-block problems.

PaLM demonstrated a performance profile akin to GPT3.5-turbo models, albeit falling short of the achievements of GPT4. Its proportion of correct plans for 3-block problems stands at 26%, but it swiftly declines to a mere 3% for 4-block tests.

Bard, a model derived from PaLM2, presented a relatively stable performance trajectory in comparison to PaLM. With a 23% correctness rate for 3-block tests, it exhibited a gradual degradation in performance as it grappled with more intricate questions.

LLaMA2 emerged as the least proficient model within the study, despite being the newest. It achieved a meager 13% correctness rate for 3-block questions and rapidly plummets to a persistent 0% correctness rate for 4-block tests and higher.

Across different prompt structures, in Natural Languages GPT4 variants performed the best out of all other LLMs, with GPT4-0613 achieving 40% validity and 20% correctness. GPT3 variants obtained a relatively high validity of 22% over the same prompt structure, but a very low correctness of 5%, which is worse than Bard and PaLM, as both models have achieved a similar validity with much higher correctness, 6% and 10% respectively. LLaMA2 continues to be the worst performing LLM, with only 6% validity and 2% validity.

In PDDL prompt structure, GPT4 variants continues to outperform other LLM counterparts, with GPT4-0613 acheiving 40% correctness and GPT4-0314 achieving 18% accuracy. GPT3 variants have decreased significally in performance and has lost its lead, reduced to 10% validity and 2% correctness. This is similar performance to PaLM and Bard, which have also suffered a validity and correctness decrease. LLaMA2, surprisingly have seen a increase in validity to 14%, surpassing GPT3 variants.

In Mystery promp structure, GPT4 variants have seen a massive performance drop, with the best performing GPT4 and GPT3-0613 only achieving 12% validity and 6% correctness. The drop in correctness was also seen in certain GPT3 variants and Bard, to only 2%. However, they have managed to maintain their validity at roughly 10%. PaLM does not seems to have be affected by the switch to Mystery domain, while LLaMA2 have failed to generate any correct plans and only have a validity of 2%.

Across various prompt structures, the GPT4 variants outperformed other Language Model (LLM) counterparts in Natural Language, with GPT4-0613 achieving a 40% validity rate and 20% correctness. On the other hand, GPT3 variants obtained a relatively high validity rate of 22% using the same prompt structures but suffered from very low correctness, only reaching 5%. This performance was notably worse than that of Bard and PaLM, as both models achieved similar validity rates with significantly higher correctness, at 8% and 10%, respectively. LLaMA2 remained the poorest performing LLM, demonstrating only 6% validity and 2

In the context of PDDL prompt structures, the GPT4 variants continued to outperform other LLM counterparts. GPT4-0613 achieved an impressive 40% correctness rate, while GPT4-0314 achieved 18% accuracy. In contrast, GPT3 variants experienced a significant decline in performance, losing their lead and achieving a mere 10% validity and 2% correctness. This level of performance was on par with that of PaLM and Bard, both of which also experienced declines in both validity and correctness. Surprisingly, LLaMA-2 witnessed an increase in validity, reaching 14% and surpassing the GPT3 variants in this specific prompt structure.

When considering the Mystery prompt structure, GPT4 variants suffered a substantial performance drop. The best-performing GPT4 and GPT4-0613 models achieved only 12% validity and 6% correctness. This drop in correctness was also observed in certain GPT3 variants and Bard, with both reaching only 2%. Nevertheless, they managed to maintain their validity at approximately 10%. PaLM appeared to be unaffected by the transition to the Mystery domain, while LLaMA-2 failed to generate any correct plans, achieving only a 2% validity rate.

5 DISCUSSION

The results obtained in this study shed light on several key aspects of LLMs and their performance in generating plans. One noteworthy finding from this study is that among the evaluated LLMs, GPT4 models performed relatively well in generating plans. However, it is essential to note that none of these models achieved excellence in plan generation. The inability to generate consistently optimal plans raises questions about the suitability of LLMs for tasks that require precise planning, especially in complex scenarios.

Another striking trend observed in the results is the downward trajectory in performance as the number of planning blocks increased. This decline in performance with increasing complexity indicates a potential limitation of current LLMs in handling intricate planning tasks. This finding underscores the importance of considering the scale and complexity of tasks when evaluating LLM performance, as it can significantly impact the outcomes.

The observed performance drop when transitioning to the Mystery domain in various LLMs raises important questions about the underlying mechanisms and capabilities of these models. It suggests that LLMs may not be engaging in substantial reasoning processes but rather relying heavily on their pre-trained data.

One possible explanation for this performance decline is the lack of extensive training data in the Mystery domain compared to the abundant planning questions available on the internet. LLMs such as GPT3 and GPT4 have been trained on vast amounts of text data from the internet, enabling them to generate contextually relevant responses in domains with which they are more familiar. However, when faced with a more "novel" domain like Mystery, where training data might be scarce or less representative, these models struggle to provide meaningful answers.

One critical issue is the prevalence of useless steps in the generated responses that do not relate to a plan, including detailed algorithms, irrelevant anecdotes involving unrelated scenarios, and even responses condemning the user's actions. For example, LLaMA2 will sometimes explain how to code a breadth-first-search algorithm in order to solve a problem, instead of providing a plan itself. Moreover, Bard will occasionally discourage the prompter from hurting animals when given a mystery domain problem.

Furthermore, while models sometimes created correct plans, they often included redundant steps that do not change the configuration state. Placing a block on the table when it is already on there is an example of a redundant step. In fact, none of the correct plans recorded were optimal. These inefficiencies in plan generation raises questions about the practicality of LLMs for real-world planning applications. Future research should focus on enhancing the ability of LLMs to generate streamlined and efficient plans that minimize unnecessary steps.

Lastly, the observation that the performance gap between LLMs narrows at higher block counts suggests a potential correlation between LLM performance and the quality of training data for lower block scenarios. This finding raises questions about the effectiveness of LLM training data in capturing planning intricacies and highlights the need for further investigation into the data selection and training processes.

6 FUTURE WORK

Our study was limited by a relatively small question set. To enhance the comprehensiveness of our findings, future work should involve a more extensive and diverse set of questions. This will allow us to explore the LLM's performance across a wider range of topics and domains.

For certain responses, the manual translation process introduced the possibility of human errors. To mitigate this concern, future research could explore the use of automated translation tools or techniques to ensure accuracy in the translation process. This would reduce the potential for bias or inaccuracies in the analysis.

One promising avenue for future research involves delving into the stochastic nature of Language Models (LLMs). Analyzing the variance among different responses generated from identical prompts can offer valuable insights into the reliability and consistency of LLMs in generating outputs. This exploration can help us better understand the factors influencing LLM behavior and provide a more robust assessment of their performance.

7 CONCLUSION

In conclusion, our examination of state-of-the-art Large Language Models (LLMs) in AI task planning, using various LLM architectures and problem complexities with the BlocksWorld planning domain, reveals that while some LLMs demonstrate relatively better performance, none consistently excel in independent plan generation. This raises doubts about their suitability for precision planning tasks, particularly in complex scenarios. Furthermore, our findings highlight a concerning decline in performance as task complexity increases, undermining the need to consider scale and intricacy when evaluating LLM capabilities. Redundant and inefficient steps in generated plans also pose practicality concerns. The narrowing performance gap between LLMs at higher complexities suggests a correlation between model performance and training data quality for simpler scenarios, warranting further investigation. While this study contributes valuable insights, future research avenues include expanding question sets, automating translation processes, and exploring LLM response variability to enhance our understanding of LLM capabilities in planning tasks, emphasizing the need for ongoing refinement in this field.

REFERENCES

[1] [n. d.]. Replicate. https://replicate.com/

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. 2022. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. https://doi.org/10.48550/arXiv.2204.01691 arXiv:2204.01691 [cs].

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. https://doi.org/10.48550/arXiv.2005.14165 arXiv:2005.14165 [cs].

[4] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling Language Modeling with Pathways. http://arxiv.org/abs/2204.02311 arXiv:2204.02311 [cs].

[5] Alfonso Gerevini and Ivan Serina. 2002. LPG: a planner based on local search for planning graphs with action costs. In *Proceedings of the Sixth International Conference on Artificial Intelligence Planning Systems (AIPS'02)*. AAAI Press, Toulouse, France, 13–22.

[6] Naresh Gupta and Dana S. Nau. 1992. On the complexity of blocks-world planning. *Artificial Intelligence* 56, 2 (1992), 223–254. https://doi.org/10.1016/0004-3702(92)90028-V

[7] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training Compute-Optimal Large Language Models. http://arxiv.org/abs/2203.15556 arXiv:2203.15556 [cs].

[8] Bin Hu, Chenyang Zhao, Pu Zhang, Zihao Zhou, Yuanhang Yang, Zenglin Xu, and Bin Liu. 2023. Enabling Intelligent Interactions between an Agent and an LLM: A Reinforcement Learning Approach. https://doi.org/10.48550/arXiv.2306.03604 arXiv:2306.03604 [cs].

[9] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., New Orleans, 22199–22213. https://papers.nips.cc/paper_files/paper/2022/file/8bb0d291acd4acf06ef112099c16f326-Paper-Conference.pdf

[10] Gary Marcus and Ernest Davis. 2020. Experiments testing GPT-3's ability at commonsense reasoning: results. https://cs.nyu.edu/~davise/papers/GPT3CompleteTests.html

[11] Drew McDermott. 1997. PDDL | The Planning Domain Definition Language. *AIPS-98 Planning Competition* (1997). https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.9941

[12] OpenAI. 2023. *GPT-4 Technical Report*. Technical Report. OpenAI, USA. https://arxiv.org/pdf/2303.08774.pdf

[13] Sundar Pichai. 2023. Bard: An important next step on our AI journey. https://blog.google/technology/ai/bard-google-ai-search-updates/

[14] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[15] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche,

Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2022. Scaling Language Models: Methods, Analysis & Insights from Training Gopher. http://arxiv.org/abs/2112.11446 arXiv:2112.11446 [cs].

[16] Tom Silver, Varun Hariprasad, Reece S. Shuttleworth, Nishanth Kumar, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2022. PDDL Planning with Pretrained Large Language Models. https://openreview.net/forum?id=1QMMUB4zfl

[17] John Slaney and Sylvie Thiébaux. 2001. Blocks World revisited. *Artificial Intelligence* 125, 1 (Jan. 2001), 119–153. https://doi.org/10.1016/S0004-3702(00)00079-5

[18] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. 2022. Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, A Large-Scale Generative Language Model. http://arxiv.org/abs/2201.11990 arXiv:2201.11990 [cs].

[19] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2023. LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models. https://doi.org/10.48550/arXiv.2212.04088 arXiv:2212.04088 [cs].

[20] Majong Studios. 2023. Minecraft. minecraft.net/en-us

[21] Kambhampati Subbarao. 2021. Language Imitation Games and the Arrival of Broad and Shallow AI. https://cacm.acm.org/blogs/blog-cacm/256068-language-imitation-games-and-the-arrival-of-broad-and-shallow-ai/fulltext

[22] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R. Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI'11)*. AAAI Press, San Francisco, California, 1507–1514.

[23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. http://arxiv.org/abs/2307.09288 arXiv:2307.09288 [cs].

[24] Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large Language Models Still Can't Plan (A Benchmark for LLMs on Planning and Reasoning about Change). https://arxiv.org/abs/2206.10498v3

[25] Karthik Valmeekam, Sarath Sreedharan, Matthew Marquez, Alberto Olmo, and Subbarao Kambhampati. 2023. On the Planning Abilities of Large Language Models (A Critical Investigation with a Proposed Benchmark). https://doi.org/10.48550/arXiv.2302.06706 arXiv:2302.06706 [cs].

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. https://doi.org/10.48550/arXiv.1706.03762 arXiv:1706.03762 [cs].

[27] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*. International Conference on Learning Representations, Kigali, Rwanda. https://arxiv.org/pdf/2203.11171.pdf

[28] Zihao Wang, Shaofei Cai, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023. Describe, Explain, Plan and Select: Interactive Planning with Large Language Models Enables Open-World Multi-Task Agents. https://arxiv.org/abs/2302.01560v1

[29] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. https://doi.org/10.48550/arXiv.2201.11903 arXiv:2201.11903 [cs].

[30] Terry Winograd. 1972. Procedures as a Representation for Data in a Computer Program for Understanding Natural Language. *Cognitive Psychology* 3, 1 (1972). https://web.archive.org/web/20200725084321/http://hci.stanford.edu/~winograd/shrdlu/AITR-235.pdf

[31] Zhenyu Wu, Ziwei Wang, Xiuwei Xu, Jiwen Lu, and Haibin Yan. 2023. Embodied Task Planning with Large Language Models. https://arxiv.org/abs/2307.01848v1

[32] Haoqi Yuan, Chi Zhang, Hongcheng Wang, Feiyang Xie, Penglin Cai, Hao Dong, and Zongqing Lu. 2023. Plan4MC: Skill Reinforcement Learning and Planning for Open-World Minecraft Tasks. https://doi.org/10.48550/arXiv.2303.16563 arXiv:2303.16563 [cs].
[33] Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, Yu Qiao, Zhaoxiang Zhang, and Jifeng Dai. 2023. Ghost in the Minecraft: Generally Capable Agents for Open-World Environments via Large Language Models with Text-based Knowledge and Memory. https://doi.org/10.48550/arXiv.2305.17144 arXiv:2305.17144 [cs].

A APPENDIX

		objects					
model type	rep	3	4	5	6	7	Overall
PaLM	NL	40	0	10	10	0	12
	My	20	10	10	0	0	8
	PD	20	0	0	0	0	4
gpt-3.5-turbo-0301	NL	30	10	0	0	0	8
	My	0	0	10	0	0	2
	PD	20	0	0	0	0	4
gpt-4-0314	NL	40	30	30	0	0	20
	My	20	10	10	0	0	8
	PD	40	40	40	0	20	28
Bard	NL	40	30	10	0	0	16
	My	10	0	0	0	0	2
	PD	20	0	0	0	0	4
gpt-3.5-turbo-0613	NL	30	0	10	0	0	8
	My	20	20	0	0	0	8
	PD	10	0	0	0	0	2
gpt-4-0613	NL	60	40	20	20	20	32
	My	30	10	10	0	0	10
	PD	30	30	20	10	0	18
gpt-3.5-turbo	NL	10	10	0	0	0	4
	My	10	0	0	0	0	2
	PD	10	0	10	0	0	4
gpt-4	NL	50	20	20	10	0	20
	My	20	10	10	0	10	10
	PD	40	20	20	0	0	16
Llama2-70b-chat	NL	20	0	0	0	0	4
	My	0	0	0	0	0	0
	PD	20	0	0	0	0	4

Table 3. Percentage of correct plans for each LLM, input structure, and number of blocks