Enhancing LLM Reasoning via Vision-Augmented Prompting

Ziyang Xiao[1], **Dongxiang Zhang**[1]*, **Xiongwei Han**[2], **Xiaojin Fu**[2], **Wing Yin Yu**[2]
Tao Zhong[2], **Sai Wu**[1], **Yuan Wang**[3], **Jianwei Yin**[1], **Gang Chen**[1]

[1] Zhejiang University [2] Huawei Noah's Ark Lab
[3] School of Business, Singapore University of Social Sciences
{xiaoziyang, zhangdongxiang, wusai, cg}@zju.edu.cn
{hanxiongwei, rocket.YuWingYin, zhongtao5}@huawei.com
fuxiaojin32@hotmail.com, Jessicawang36@gmail.com
zjuyjw@cs.zju.edu.cn

Abstract

Verbal and visual-spatial information processing are two critical subsystems that activate different brain regions and often collaborate together for cognitive reasoning. Despite the rapid advancement of LLM-based reasoning, the mainstream frameworks, such as Chain-of-Thought (CoT) and its variants, primarily focus on the verbal dimension, resulting in limitations in tackling reasoning problems with visual and spatial clues. To bridge the gap, we propose a novel dual-modality reasoning framework called Vision-Augmented Prompting (VAP). Upon receiving a textual problem description, VAP automatically synthesizes an image from the visual and spatial clues by utilizing external drawing tools. Subsequently, VAP formulates a chain of thought in both modalities and iteratively refines the synthesized image. Finally, a conclusive reasoning scheme based on self-alignment is proposed for final result generation. Extensive experiments are conducted across four versatile tasks, including solving geometry problems, Sudoku, time series prediction, and travelling salesman problem. The results validate the superiority of VAP over existing LLMs-based reasoning frameworks.

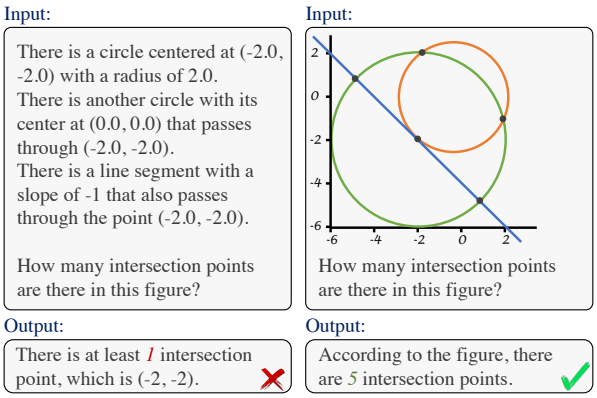
1 Introduction

The human cognitive system is characterized by the presence of two specialized subsystems within the working memory: the phonological loop, which processes verbal information, and the visual-spatial sketchpad, which processes visual and spatial information [1]. Both of them play a crucial role in problem-solving by offering qualitatively distinct strategies for comprehending and manipulating information [2]. Recently, with the rapid advancement of LLMs, there have emerged various reasoning frameworks, such as Chain of Thought (CoT) [3], Self-consistent CoT (CoT-SC) [4], and Tree of Thoughts (ToT) [5]. Although these frameworks have shown impressive performance across a wide range of NLP tasks, they primarily focus on the verbal dimension with text-only representations, resulting in limitations in tasks that require visual and spatial interpretation (e.g., geometry problems or grid puzzles).

In this paper, we propose a novel dual-modality reasoning approach called **V**ision-**A**ugmented **P**rompting (VAP) that analogizes human cognition subsystems with the assistance of multimodal large language models (MLLMs). VAP takes textual problems as input and uses self-synthesized images as an additional information channel to enhance reasoning. As shown in Figure 1, we employ

*Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) With text-only input (b) Enhanced with visual input

Figure 1: An example of solving the same Geometry Intersection Counting problem using different types of input. Outputs are derived from GPT-4; for brevity, only the conclusions are presented.

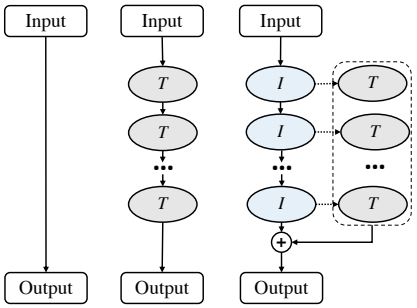


Figure 2: Compare with Standard Prompting and Chain of Thoughts Prompting(CoT).

the state-of-the-art GPT-4V(ision) [6] to solve a geometry problem. In this case, the model accurately deduces the correct answer when given both image and text inputs. In contrast, it generates an ambiguous answer with purely textual information. This example is analogous to the process of human cognition where it is a common practice to use images to enhance comprehension when handling geometry problems [7].

Our proposed vision-augmented prompting comprises three steps. Initially, LLMs generates a high-level plan for the following steps, including selecting an appropriate drawing toolkit and creating an initial image. To automate the procedure, we leverage the API documentation of external tools as context of LLM to facilitate drawing tool selection and figure synthesis. In the second step, VAP iteratively performs reasoning on the image, updates it, and generates an accompanying textual thought in each iteration. This process results in a chain of thoughts in both image and text modalities, as illustrated in Figure 2. Lastly, the final image and the chain of textual thoughts are jointly fed to the MLLM to derive a conclusive answer. To enhance robustness, we introduce a technique called self-alignment, where the MLLM describes the image content first, and the image channel is discarded if the self-description fails to align with the initial high-level plan.

We conduct extensive experiments across four versatile tasks, among which VAP establishes new state-of-the-art performance compared to other training-free LLMs-based methods. These four reasoning tasks include: (1) Geometry Intersection Counting in the domain of geometry word problems (+5.0% absolute accuracy gains); (2) Sudoku Puzzle as a logical reasoning task (+12.9%); (3) Time Series Prediction as a numerical analysis task (+9.9% in reducing mean absolute error); and (4) Travelling Salesman Problem as a classical NP-hard problem in the field of operations research (+1.8% in reducing the optimal gap).

2 Related Work

2.1 LLMs-based Reasoning

Improving the reasoning capabilities of LLMs such as GPT-4 [8], PaLM2 [9], and LLaMA2 [10] has been a hot topic in recent years. To achieve the goal, Chain-of-Thought (CoT) [3] breaks a reasoning task into a series of intermediate reasoning steps. Self-consistency [4] replaces the naive greedy decoding in CoT by sampling a diverse set of reasoning paths and selecting the most consistent answer. Tree of Thoughts (ToT) [5] and Graph of Thoughts (GoT) [11] further allow LLMs to explore and combine thoughts in a structured manner. Cumulative Reasoning (CR) [12] decomposes a reasoning task into smaller components, which are solved in a cumulative and iterative manner. Additionally, Chain-of-Experts [13] enhances reasoning capabilities through the collaboration of multiple LLMs.

2.2 Multimodal Large Language Models

The emergence of Multi-modal Large Language Models (MLLMs) such as GPT-4V(ison) [6] has fostered a new research landscape. Although the architecture of GPT-4V is not publicly available, substantial progress has been made in the open-source domain [14, 15, 16, 17]. These works usually fine-tune traditional text-based LLMs to align with other modalities. Similar approaches are also adopted by multimodal chatbots [18] and multimodal universal task solvers [15, 19] for vision-related tasks [20, 21, 22].

2.3 MLLMs-based Reasoning

According to a recent survey [23], reasoning based on MLLMs can be broadly categorized into LLM-aided visual reasoning and multi-modal Chain-of-Thought. LLM-aided visual reasoning focuses on solving traditional visual reasoning tasks with the assistance of LLMs. This includes tasks such as visual question answering [24], image segmentation [25], and video question answering [26]. Some works in this category, like ViperGPT [27], VisProg [28], and LLava-Plus [29], also use MLLMs to call external tools for enhanced reasoning, while their task settings differ from ours. On the other hand, the multi-modal Chain-of-Thought extends traditional prompting techniques to the multi-modal context. MM-ReAct [30] extends ReAct [31] to support multimodal data. The Visual Chain of Thought (VCoT) [32] uses CoT with vision-language grounding to bridge the gap in multi-step temporal reasoning. Compared with our work, VCoT is designed to bridge the logical gaps within sequential data and facilitate temporal reasoning in tasks like visual storytelling and WikiHow summarization.

3 Methodology

Mainstream LLMs-based reasoning frameworks, such as CoT, ToT, and GoT, only consider the verbal domain $\mathcal{L}$. To tackle more challenging problems with visual and spatial clues, we propose vision-augmented prompting (VAP) and extend the exploration space from a single domain $\mathcal{L}$ to a dual-modality domain $\mathcal{L} \cup \mathcal{G}$, where $\mathcal{G}$ represents the visual-spatial domain. It relies on the external image synthesis toolkit to automatically draw an image matching the text description of the input problem (Section 3.1). The core challenge lies in how to effectively navigate the joint space and maximize the success rate of problem solving, which will be presented in Section 3.2.

3.1 External Image Synthesis Toolkit

While visual-spatial information can be beneficial for reasoning as demonstrated in Figure 1, LLMs lack the inherent ability to visualize concepts. Therefore, it is necessary to leverage external image synthesis toolkit. Vision-augmented prompting incorporates graphic rendering tools that rely on logical programming to render images, including Python Turtle[2] for graphical visualizations, and Matplotlib[3] for drawing analytical figures. In addition to these third-party programmable tools, we further integrate image generative models, such as DALL·E 3 [33], to produce images directly from textual prompts. Consequently, we define the set of our drawing tools as $\mathcal{S}_T = \{\text{Turtle}, \text{Matplotlib}, \text{DALL·E 3}\}$. These tools will be utilized by our method through API calls.

3.2 Procedures of Vision-Augmented Prompting

We model the problem-solving process of VAP as an iterative reasoning process. With the aid of the image synthesis tools, VAP progressively updates the image according to the instruction provided by the language model. To maintain a coherent reasoning trajectory, a 'thought' is generated on each iteration. Subsequently, to derive the conclusive answer, the final image, the original problem, and the trajectory of iterative thoughts are sent to the MLLM to obtain the final answer. However, we find it challenging for LLMs to perform iterative multi-step drawing and reasoning directly due to the lack of a global view. To overcome this limitation, we introduce a planning step, where LLMs will create a high-level plan for subsequent steps. Furthermore, it is possible that the synthesized image could be disqualified and mislead the reasoning process. To alleviate the negative effect of such images, we

[2] `https://pythonturtle.org/`
[3] `https://matplotlib.org/`

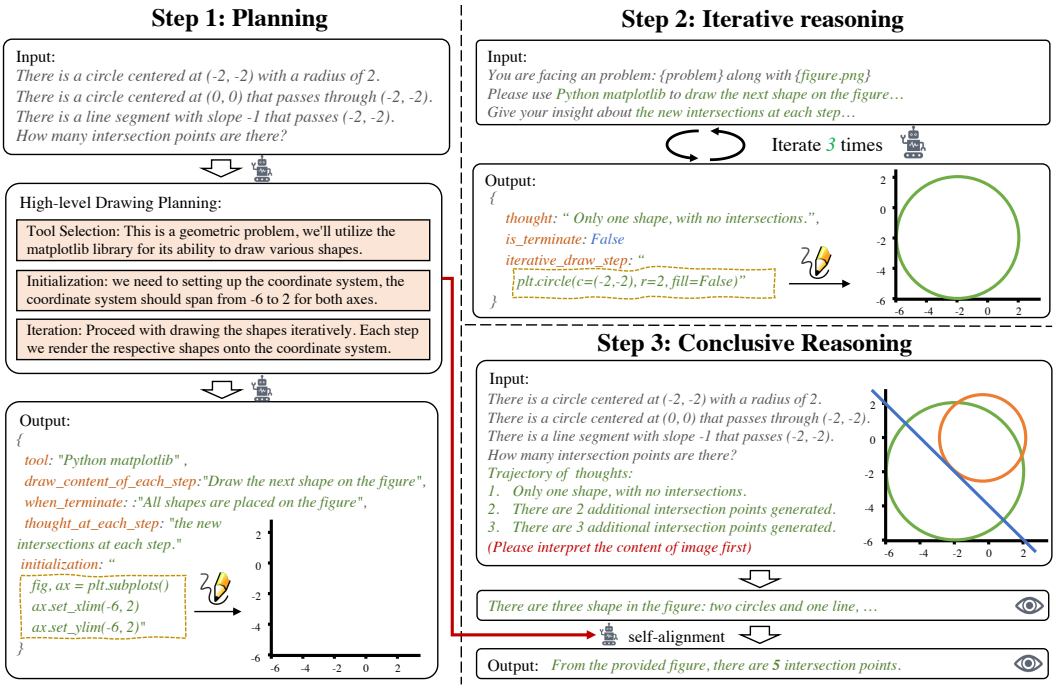


Figure 3: Illustration of the workflow in VAP.

introduce a technique named self-alignment to enforce MLLMs to literally describe the synthesized image and check whether its text description is aligned with the initial high-level plan. If disparity is detected, we discard the image and restart the reasoning process.

Step 1: Planning As demonstratrd in Figure 3, our method takes a textual problem $\mathcal{P}$ as input and firstly generates a high-level reasoning plan. In implementation, we first prompt the LLM to acquire a detailed description of the plan in natural language. The output of the plan consists of five key components: (1) an analysis of the problem's characteristics for visualization, along with the chosen drawing tool T from the set of available tools $\mathcal{S}_T$; (2) A description of how to initialize the image using the chosen tool, accompanied by the corresponding API call instruction I_0; (3) the prompt $\mathbb{P}_d$ that drives the LLM to draw in each iterative step; (4) the prompt $\mathbb{P}_t$ that drives the LLM to think step by step during the iterative drawing process; (5) the termination condition C for the iteration. To facilitate subsequent reasoning, we further transform the unstructured textual description of the planning into semi-structured JSON format. This step can be formalized as shown in Equation 1, where $\mathcal{F}_{planning}$ denotes the language model involved in this step with prompt engineering. With the instruction I_0, we can create the initial image $g_0 \in \mathcal{G}$ using the selected tool.

$$\{T, I_0, \mathbb{P}_d, \mathbb{P}_t, C\} \leftarrow \mathcal{F}_{planning}(\mathcal{P}, \mathcal{S}_T),\ g_0 \leftarrow T(I_0) \tag{1}$$

Note that both plan generation and JSON format transformation are implemented using LLMs with prompt engineering. Details of the prompt templates can be found in Appendix A.1.1.

Step 2: Iterative reasoning In the iterative reasoning step, the MLLM is specified by two prompts $\mathbb{P}_d$ and $\mathbb{P}_t$ generated in the previous step, along with the image synthesis tool $\mathcal{S}_T$ and the assigned termination condition C. We denote the MLLM in this step as $\mathcal{F}_{\{\mathbb{P}_d, \mathbb{P}_t, \mathcal{S}_T, C\}}$. In each iteration t, the MLLM takes the original problem $\mathcal{P}$, the partially-completed image g_t and trajectory of thoughts Z_t as input. The MLLM will then generate an instruction I_t containing API calls used to update the image g_t. Regarding the update of the image, an accompanying 'thought' z_t that provides a textual interpretation of the current state is generated. The process is formally depicted in Equation 2, with details of the prompt template presented in Appendix A.1.2. Here, Z_t starts as an empty array and the new 'thought' z_t is appended to the trajectory at each iteration.

$$\{z_t, I_t\} \leftarrow \mathcal{F}_{\{\mathbb{P}_d, \mathbb{P}_t, \mathcal{S}_T, C\}}(\mathcal{P}, g_t, Z_t),\ Z_{t+1} \leftarrow Z_t \cup \{z_t\},\ g_{t+1} \leftarrow T(I_t) \tag{2}$$

We observe that the MLLM occasionally failed to follow the instructions provided in the prompt (e.g., repeatedly drawing the same shape), causing disruption to the entire iterative process. To address this issue and improve the stability of reasoning process, we enrich the context of MLLM by appending the input and output from the last step, which serve as illustrative examples for the MLLM to better follow the instructions.

Step 3: Conclusive reasoning When the iterative reasoning terminates (i.e., condition C is met), we proceed to the final step of conclusive reasoning using the synthesized image of step 2, denoted by g_n. Since we cannot guarantee g_n is a perfect output and a disqualified g_n may mislead the conclusive reasoning process, we introduce a technique named self-alignment to enforce MLLMs to literally describe the synthesized image and check whether its text description is aligned with the initial high-level plan. If disparity is detected, we discard the image and restart the reasoning process. If a qualified image cannot be obtained after a certain number of trials, VAP is degraded to simple input-output reasoning, without leveraging the visual channel.

As shown Equation 3, our conclusive reasoning takes g_n, the trajectory of thoughts Z_n, along with the original problem $\mathcal{P}$ as input and leverage the prompt template presented in Appendix A.1.3 to derive the final answer $\mathcal{A}$.

$$\mathcal{A} \leftarrow \mathcal{F}_{self-alignment}(\mathcal{P}, g_n, Z_n) \tag{3}$$

4 Experiments

We evaluate VAP using four diversified and challenging tasks, including Geometry Intersection Count, Sudoku Puzzle, Time Series Prediction, and Travelling Salesman Problem. We also provide a typical input-output example for each task in Appendix A.2.

These tasks share three LLM-based reasoning baselines, including standard prompting, chain-of-thought (CoT) prompting, and CoT with self-consistency (CoT-SC) prompting [4].

For CoT prompting, the reasoning process involves multiple intermediate steps. In the Geometry Intersection Counting task, we define each intermediate step as calculating the number of intersections between a pair of shapes. In the Sudoku Puzzle task, each step involves considering a position that violates the Sudoku rules. For Time Series Prediction and Travelling Salesman Problem, it is challenging to define specific step content. Therefore, we append a 'think step by step' instruction to the standard prompt and include an additional step to extract the solution from the generated thoughts.

Additionally, for CoT-SC prompting, when the task output is discrete (i.e., Geometry Intersection Counting, Sudoku Puzzle, and Travelling Salesman Problem), we sample k answers and use the majority answer. When the task output is continuous (i.e., numeric value in the task of Time Series Prediction), we generate k predictions and calculate their average as the final forecast value. The default k is set to 10.

For fairness, we employ the 'GPT-4-vision-preview' as the underlying MLLM for these baselines and our VAP. The default temperature is set to 0. For methods that require sampling, such as SC, the temperature is set to 0.7.

4.1 Task 1: Geometry Intersection Counting

Task Description The task determines the number of intersection points between a couple of geometric shapes described in natural language. For example, given the input "There is a line segment from $(-2.5, -1)$ to $(-0.5, -1)$ and a circle centered at $(-1.5, -1)$ with radius 1," the correct output should be "2", as there are 2 intersection points between the circle and the line segment.

Task Setup We randomly sample 200 problem instances from Geometry Intersection Counting task in the BIG-bench benchmark[4]. The performance metric used in this task is accuracy, which is defined as the percentage of problem instances in which the output number matches the ground truth.

[4] `https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks/intersect_geometry`

Task-specific Baselines We introduce two additional baselines specifically developed for solving geometry problems. The first is Inter-GPS [34], a symbolic reasoner [35] for geometry problems. The second is G-LLaVA [36], a MLLM specialized trained on geometry problems. Since geometry figures are not available in the problem input, G-LLaVA only leverages the textual modality for reasoning.

Table 1: Intersection Point Result.

Method	Accuracy
Standard	8.5%
CoT	10.0%
CoT-SC$_{(k=5)}$	11.0%
CoT-SC$_{(k=10)}$	11.5%
CoT-SC$_{(k=20)}$	11.5%
Inter-GPS	6.0%
G-LLaVA	14.0%
VAP	**16.5%**

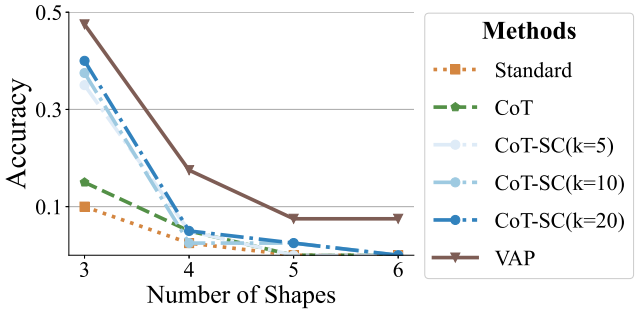


Figure 4: Accuracy with different number of shapes.

Results As shown in Table 1, Inter-GPS and G-LLaVA exhibit low accuracy, primarily because they are originally trained on dual-modality datasets and thus show limited effectiveness with text-only input. We also examine their performance when provided with an additional image input in Section 4.5, where the purpose is to validate the effectiveness of the synthesized images by our VAP algorithm. Among the general-purpose reasoning methods, standard prompting and CoT prompting yield relatively low accuracy (8.5% and 10.0% respectively). The CoT-SC method, even with a sample size of 20, only marginally improves accuracy by 1.5%. In contrast, VAP significantly outperforms these methods and achieves an accuracy of 16.5%. Furthermore, in the break-down analysis with increasing number of shapes, as illustrated in Figure 4, VAP demonstrates clear superiority over other LLM-based methods in more complex scenarios involving four or more shapes. We can see that the accuracy of all baselines is almost close to 0 in these complex scenarios.

4.2 Task 2: Sudoku Puzzle

Task Description In this task, an initial state of a Sudoku board is presented in natural language, with digits on filled cells and dots denoting empty cells. For example, in a 9×9 board, a row "..6 ... ..." only contains digit 6 in the third column. The reasoning process iteratively provides the information of the next action, in the form of "x y digit", where 'x' and 'y' specify the cell's coordinates on the board, and 'digit' is the number to be placed in that cell.

Task Setup We utilize the Sudoku puzzle generation program from the BIG-bench[5] to create a dataset that includes 150 Sudoku puzzles. For performance evaluation, we employ correct rate and collision rate as two metrics. The correct rate measures the accuracy in solving the puzzles, and collision rate assesses the frequency at which the position in given command is already filled with numbers, which indicates a violation of Sudoku rules.

Task-specific Baselines For a more comprehensive evaluation, we also incorporate Tree of Thought (ToT) [37] as an additional baseline. The implementation of ToT in this task is not troublesome because it is straightforward to decompose the thought process into a tree-structured representation for board games.

Results In Table 2, the standard prompting shows a low success rate of 18.0%, with a high rate of rule violations. This can be attributed to Sudoku's demands for reasoning and rule comprehension, which pose a challenge for LLMs relying solely on prompt engineering. The CoT and CoT-SC methods show improved performance, as they can, to certain extent, enhance the coherence of LLM reasoning by providing a reference of previous steps. The ToT algorithm is the most effective among these baselines, achieving a 22.6% success rate, owing to its tree-structured thought process. Notably,

[5] https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks/sudoku

VAP outperforms all other methods with a correct rate of 35.5%. VAP also reduces the collision rate, implying that integrating the image modality is helpful for LLMs to understand game rules.

Table 2: Sudoku Result.

Method	Correct rate↑	Collision rate↓
Standard†	18.0%	68.7%
CoT	17.3%	64.7%
CoT-SC$_{(k=5)}$	20.6%	48.7%
CoT-SC$_{(k=10)}$	20.6%	47.3%
ToT	22.6%	44.0%
VAP	**35.5%**	**26.0%**

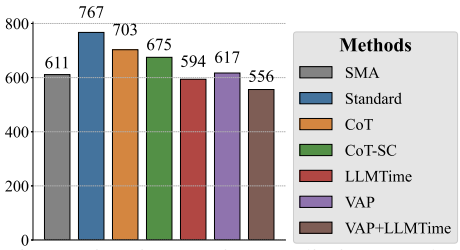


Figure 5: Time series prediction results, with y-axis indicating MAE.

4.3 Task 3: Time Series Prediction

Task Description To evaluate the numerical analysis capabilities, we regard the LLM as a time series predictor. In this task, a sequence of data points is provided, and the objective for the LLM is to predict the next n values in the series.

Task Setup The dataset for this task is sourced from the Darts library [38], which includes a curated collection of 8 real univariate time series datasets. We configure the window size to 100 data points and predict the subsequent $n = 8$ values in the series. The performance is evaluated using the Mean Absolute Error (MAE) metric.

Task-specific Baselines We chose the LLMTime algorithm [39] as the most cutting-edge method for comparison, where a data rescaling strategy and a tokenization trick is introduced to enhance numerical precision. For instance, the sequence "8.05, 1, 35" will be tokenized as "8 0 5 , 1 0 0 , 3 5 0 0". We also adopt the simple moving average (SMA) as a traditional statistical baseline for time series forecasting.

Results As shown in Figure 5, SMA is recognized as a traditionally effective method for time series prediction, surpassing most LLM-based methods with an MAE of 611. Among the LLM-based approaches, compared to standard prompting, CoT-SC approach significantly reduces the MAE from 767 to 675, justifying the effectiveness of integrating auto-regression and sample strategy. It is noteworthy that LLMTime, as a preprocessing technique for LLMs to handle sequential data, performs effectively. Its performance even marginally exceeds that of SMA. In the result, VAP outperforms the majority of LLM-based methods and is slightly inferior to LLMTime. However, considering that VAP is orthogonal to LLMTime, we introduce a new method that combines the same preprocessing technique with VAP. The results show that this combined version, VAP+LLMTime, outperforms all comparison algorithms.

4.4 Task 4: Travelling Salesman Problem

Next, we test our method on travelling salesman problem (TSP), an NP-hard combinatorial optimization problem that poses challenges for neural network solvers [40]. The input for TSP instances consists of a set of coordinates, and the output is a sequence of city indices representing the route for the salesman.

Task Setup We use Euclidean TSP instances with 10 and 20 cities as our testset. For each city size, 100 instances are generated by a program, with coordinates uniformly distributed within a $[0, 100]^2$ integer grid. The average path length and the gap from the optimal solution are used as metrics.

Task-specific Baselines We introduce four traditional TSP solvers as our task-specific baselines: Gurobi, an exact TSP solver using mixed integer programming; Nearest neighbour (NN) algorithm, which greedily selects the closest city; Fastest insertion (FI), an efficient insertion method; and a Random baseline that chooses paths randomly for performance benchmarking. Here, note that the

output of LLM-based methods (i.e., Standard, CoT, CoT-SC, and VAP) might violate constraints of TSP (e.g., generating duplicate cities). In such cases, we randomly select an unvisited city to continue the solution.

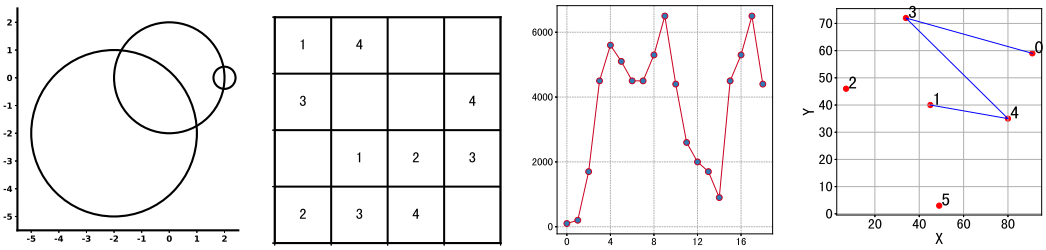
Table 3: TSP task performance.

	Method	Traditional				LLMs-based			
		Gurobi	Random	NN	FI	Standard	CoT	CoT-SC	VAP
N=10	**Length**	**294.8**	529.6	327.4	306.8	327.5	327.2	325.0	**312.4**
	Gap	**0.0%**	79.7%	11.1%	4.1%	11.1%	11.0%	10.4%	**6.0%**
N=20	**Length**	**383.7**	989.5	448.2	424.2	544.5	547.2	541.9	**499.3**
	Gap	**0.0%**	170.7%	16.8%	10.6%	41.9%	42.6%	41.2%	**30.2%**

Results As shown in Table 3, the performance of standard prompting is comparable to the Nearest Neighbour algorithm when applied to TSP with 10 cities. CoT demonstrates very slight improvement over standard prompting. This might be attributed to the TSP's requirement for a heuristic and complex thought process, which is challenging for CoT to replicate. VAP outperforms all LLM-based algorithms, achieving a 6.0% gap for optimality, close to the Fastest Insertion's performance. It's worth noting that when the number of cities in a problem instance increases to 20, the performance of LLM-based algorithms significantly decreases as the search space of TSP grows exponentially with the problem size. Such a huge search space poses a significant challenge for LLMs not specifically tailored to solve TSP. Nonetheless, VAP's superiority over other LLM-based methods is still evident when the problem size increases.

4.5 Effectiveness of the Synthesized Images

To demonstrate the quality of synthesized images, we first present several illustrative examples in Figure 6 and then provide numerical experiments in the following.



(a) Geometry example (b) Sudoku example (c) Time series example (d) TSP example

Figure 6: Examples of images synthesized by our VAP among the four tasks.

In the first experiment to evaluate the quality of synthesized images, we use the rate of integrity as the metric, which refers to whether the generated image contains all the elements described in the original problem. This is a relatively objective metric for human annotators to reach consensus. As shown in Table 4, the images synthesized by VAP demonstrate a high integrity rate across various tasks. Notably, for Time Series Prediction and TSP, the module achieves an impressive integrity rate of 100% and 98.0%, respectively. We also evaluate the performance improvement of VAP when provided with the ground truth image. As shown in Table 4, the column of 'With ground truth image' refers to the performance boost rate. If the drawing errors had been corrected, we can observe notable potential for improvement. These results underscore the crucial role of accurate image rendering in enhancing the final output quality of the VAP.

In the next experiment, we provide the synthesized images by our VAP as additional input for the geometry solvers Inter-GPS and G-LLaVA involved in task of Geometry Intersection Counting. As shown in Table 5, these two approaches can greatly benefit from our synthesized images. Their remarkable accuracy boosting validates the effectiveness of the images generated by VAP.

Table 4: Performance of Drawing Module.

Task	Integrity	With ground truth image
Geometry	83.5%	5.8% (17.0% → 18.0%)
Sudoku	89.3%	5.1% (35.5% → 37.3%)
Time Series	100.0%	-
TSP	98.0%	1.3% (30.2% → 30.6%)

Table 5: Performance of dedicated geometry solvers with (denoted by †) and without synthesized images.

Method	Accuracy	Accuracy†
Inter-GPS	6.0%	**17.0%**
G-LLaVA	14.0%	**20.5%**

5 Ablation Study

In the ablation study, we evaluate the effectiveness of three steps, including high-level planning, iterative reasoning and self-alignment in conclusive reasoning. Note that removing the planning step refers to prompting the LLM to draw the image step-by-step without providing a plan in advance. Removing iterative reasoning refers to directly prompting the LLM to generate an entire image and then proceeding to the conclusive reasoning step without intermediate results.

As shown in Table 6, we report the core metric for four tasks: accuracy for Geometry Intersection Problems, correct rate for Sudoku puzzles, MSE for Time Series Prediction, and average tour length for the TSP with 10 cities. The results demonstrate that removing any of the key components causes a performance degradation, except for self-alignment in Time Series Prediction. This is because there is no error in the drawing module, as discussed in Subsection 4.5. Hence, in this scenario, self-alignment plays no effect. Interestingly, removing planning step causes the most significant performance drop for Geometry Intersection Problems and TSP. The reason is that planning is crucial for visualizing problems requiring precise spatial relationships. Furthermore, iterative reasoning proves to be an essential component, as its removal leads to considerable performance degradation across all tasks. This finding highlights the importance of our view that all problem-solving is a step-by-step process. Self-alignment plays an important role in the task of Sudoku. For example, the LLM will occasionally draw the incorrect coordinate range for the Sudoku board, resulting in a board with the wrong size. However, with self-alignment, the model first describes the board size in the figure and cross-checks this against the original plan, preventing such issues.

Table 6: Ablation study results. Each column represents a task and the cell indicates the performance when removing a module in VAP.

	Geometry	Sudoku	Time Series	TSP$_{(N=10)}$
VAP	16.5%	35.5%	556	312.4
w/o planning	12.0%	24.7%	578	322.0
w/o iterative	14.5%	19.9%	582	321.3
w/o self-alignment	16.0%	25.1%	556	315.9

6 Limitations of VAP

Even though VAP has been shown to outperform other LLM-based approaches in a set of versatile tasks, there still exists a noticeable performance gap between VAP and task-specific approaches. However, these tailored approaches require nontrivial efforts for domain customization and is not able to support other types of reasoning tasks. In contrast, VAP is a lightweight and training-free framework. With negligible customization cost, VAP is general enough to handle a spectrum of complex reasoning tasks with visual and spatial clues.

The other limitation is the black-box functionality of VAP, even though the framework is inspired by the cooperation of two subsystems in different regions of human brain that cooperate with each other. In Appendix A.3, we provide two case studies as examples to provide certain insights and justify the effectiveness of VAP. It would be an interesting future research direction to explore the interpretability of VAP and its relationship with cognitive science.

7 Conclusion

In this paper, we introduced a novel reasoning framework called Vision-Augmented Prompting (VAP), designed to enhance the reasoning capabilities of large language models (LLMs) by emulating the human cognitive system's dual modality processing. VAP leverages the external image synthesis tools to generate visual representations that augment the textual input. The procedure of VAP follows a three-step algorithm: first, it automatically generates a high-level plan for reasoning; second, it engages in iterative drawing and reasoning based on the partially-completed image; and finally, original problem, all thoughts and generated image are jointly to derive a solution. To enhance robustness, we introduced a self-alignment technique, where the MLLM describes the image content, and the image channel is discarded if the self-description fails to align with the initial high-level plan. We conducted extensive experiments across four diverse reasoning tasks: intersection counting in Geometry Intersection Problems, Sudoku Puzzles, Time Series Prediction, and the Travelling Salesman Problem. The results demonstrated the effectiveness of VAP, establishing new state-of-the-art performance compared to other training-free LLM-based methods.

Acknowledgments and Disclosure of Funding

This work is supported by the National Key Research and Development Project of China (2022YFF0902000) and the Fundamental Research Funds for the Central Universities (226-2024-00145 and 226-2024-00216).

References

[1] Alan Baddeley. Working memory. *Science*, 255(5044):556–559, 1992.

[2] Mary Hegarty Maria Kozhevnikov and Richard E. Mayer. Revising the visualizer-verbalizer dimension: Evidence for two types of visualizers. *Cognition and Instruction*, 20(1):47–77, 2002.

[3] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[4] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[5] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *CoRR*, abs/2305.10601, 2023.

[6] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision), 2023.

[7] Judith E Fan, Wilma A Bainbridge, Rebecca Chamberlain, and Jeffrey D Wammes. Drawing as a versatile cognitive tool. *Nature Reviews Psychology*, 2(9):556–568, 2023.

[8] OpenAI. Gpt-4 technical report, 2023.

[9] Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas

Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. Palm 2 technical report, 2023.

[10] 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. Llama 2: Open foundation and fine-tuned chat models, 2023.

[11] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. *CoRR*, abs/2308.09687, 2023.

[12] Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. Cumulative reasoning with large language models. *CoRR*, abs/2308.04371, 2023.

[13] Ziyang Xiao, Dongxiang Zhang, Yangjun Wu, Lilin Xu, Yuan Jessica Wang, Xiongwei Han, Xiaojin Fu, Tao Zhong, Jia Zeng, Mingli Song, and Gang Chen. Chain-of-experts: When llms meet complex operations research problems. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[14] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *CoRR*, abs/2304.08485, 2023.

[15] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven C. H. Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *CoRR*, abs/2305.06500, 2023.

[16] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *CoRR*, abs/2303.16199, 2023.

[17] Tao Gong, Chengqi Lyu, Shilong Zhang, Yudong Wang, Miao Zheng, Qian Zhao, Kuikun Liu, Wenwei Zhang, Ping Luo, and Kai Chen. Multimodal-gpt: A vision and language model for dialogue with humans. *CoRR*, abs/2305.04790, 2023.

[18] Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. Minigpt-v2: large language model as a unified interface for vision-language multi-task learning, 2023.

[19] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, and Jifeng Dai. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *CoRR*, abs/2305.11175, 2023.

[20] Wentao Tan, Changxing Ding, Jiayu Jiang, Fei Wang, Yibing Zhan, and Dapeng Tao. Harnessing the power of mllms for transferable text-to-image person reid. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 17127–17137. IEEE, 2024.

[21] Zepeng Li, Dongxiang Zhang, Yanyan Shen, and Gang Chen. Human-in-the-loop vehicle reid. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 6048–6055. AAAI Press, 2023.

[22] Zepeng Li, Dongxiang Zhang, Sai Wu, Mingli Song, and Gang Chen. Sampling-resilient multi-object tracking. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 3297–3305. AAAI Press, 2024.

[23] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*, 2023.

[24] Haoxuan You, Rui Sun, Zhecan Wang, Long Chen, Gengyu Wang, Hammad A. Ayyubi, Kai-Wei Chang, and Shih-Fu Chang. Idealgpt: Iteratively decomposing vision and language reasoning via large language models, 2023.

[25] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. *arXiv preprint arXiv:2308.00692*, 2023.

[26] Antoine Yang, Antoine Miech, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Zero-shot video question answering via frozen bidirectional language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[27] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 11854–11864. IEEE, 2023.

[28] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training, 2022.

[29] Shilong Liu, Hao Cheng, Haotian Liu, Hao Zhang, Feng Li, Tianhe Ren, Xueyan Zou, Jianwei Yang, Hang Su, Jun Zhu, Lei Zhang, Jianfeng Gao, and Chunyuan Li. Llava-plus: Learning to use tools for creating multimodal agents. *CoRR*, abs/2311.05437, 2023.

[30] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. MM-REACT: prompting chatgpt for multimodal reasoning and action. *CoRR*, abs/2303.11381, 2023.

[31] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[32] Daniel Rose, Vaishnavi Himakunthala, Andy Ouyang, Ryan He, Alex Mei, Yujie Lu, Michael Saxon, Chinmay Sonar, Diba Mirza, and William Yang Wang. Visual chain of thought: Bridging logical gaps with multimodal infillings, 2023.

[33] Zhan Shi, Xu Zhou, Xipeng Qiu, and Xiaodan Zhu. Improving image captioning with better use of captions, 2020.

[34] Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6774–6786. Association for Computational Linguistics, 2021.

[35] Ziyang Xiao and Dongxiang Zhang. A deep reinforcement learning agent for geometry online tutoring. *Knowl. Inf. Syst.*, 65(4):1611–1625, 2023.

[36] Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, and Lingpeng Kong. G-llava: Solving geometric problem with multi-modal large language model, 2023.

[37] Jieyi Long. Large language model guided tree-of-thought, 2023.

[38] Julien Herzen, Francesco Lässig, Samuele Giuliano Piazzetta, Thomas Neuer, Léo Tafti, Guillaume Raille, Tomas Van Pottelbergh, Marek Pasieka, Andrzej Skrodzki, Nicolas Huguenin, Maxime Dumonal, Jan Koscisz, Dennis Bader, Frédérick Gusset, Mounir Benheddi, Camila Williamson, Michal Kosinski, Matej Petrik, and Gaël Grosch. Darts: User-friendly modern machine learning for time series. *J. Mach. Learn. Res.*, 23:124:1–124:6, 2022.

[39] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. *CoRR*, abs/2310.07820, 2023.

[40] Dongxiang Zhang, Ziyang Xiao, Yuan Wang, Mingli Song, and Gang Chen. Neural TSP solver with progressive distillation. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 12147–12154. AAAI Press, 2023.

[41] Richard C Larson and Amedeo R Odoni. *Urban operations research*. Number Monograph. 1981.

A Appendix

A.1 Prompt Engineering of VAP

A.1.1 Planning

The planning step's prompt is divided into two sub-steps. First, a high-level plan's textual representation is generated using the following prompt. Here, the '{examples}' and '{problem}' are placeholders for the few-shot examples (which will be discussed later) and the problem input, respectively.

> Your role is to visualize a problem by creating an image that represents the problem description accurately using a specific tool. This task involves drawing an image that encapsulates all the details provided in the problem description.
> The drawing will be executed through an iterative process. This means you will develop the image in a step-by-step manner, ensuring that each element of the problem is represented accurately.
> Before beginning the drawing, you are required to outline a high-level plan for how you will approach the drawing process. This plan should cover three essential aspects:
> 1. Tool Selection - The choice of software or tool you will use for drawing. Here are drawing tools available:
> - Python Matplotlib: a powerful Python library used for creating a wide variety of static, animated, and interactive visualizations. It is widely utilized in data science and engineering for generating high-quality plots and graphs.

//=== Continued from previous page ===
- Turtle: a Python library that provides a simple way to draw graphics and shapes using a virtual "turtle" that moves around the screen. Inspired by the Logo programming language, it is an excellent tool for teaching programming concepts through visual feedback.
- DALLE3: an advanced image generation model, capable of creating detailed and imaginative images from textual descriptions. It leverages deep learning to understand and produce highly realistic or fantastical scenes based on user prompts.
2. Initialization Approach: How you will begin your drawing, focusing on setting up the initial state of the image.
3. Iterative Drawing Approach: A detailed description of how you will iteratively add details to the image step by step.
Output Format: Your plan should be organized into three distinct paragraphs, each starting with the respective headings: Tool selection:, Initialization:, and Iteration:.
{examples}
Problem Description: {problem}

The few-shot examples are optional to improve the LLM's ability to handle various tasks. The planning step plays a crucial role in the overall reasoning process, therefor, we use a one-shot example as our default setting (the same as other baselines for fairness). For the planning step, we use several pairs of problem input and textual high-level plan as examples. For other standard prompting, we use the problem input and answer output as examples. For CoT prompting and CoT-SC prompting, we use problem input, trajectory of thoughts, and answer output as examples.

Subsequently, a semi-structured plan in JSON format is extracted using the following prompt. Here, {drawing_plan} is the placeholder of the high-level plan generated before.

You are tasked with visualizing problems by creating images. Each image should be constructed step by step, based on a detailed plan you've previously prepared.
You should first review your plan, and start by examining the high-level plan you've made for the iterative drawing process.
Your plan is as following: {drawing_plan}.
Your task is to extract meta-information from your drawing plan and format it as a JSON string.
Follow this strict JSON structure in your output:
{
"tool": "Name of the tool you're using",
"draw_content_of_each_step": "Describe what you'll draw in each step",
"when_terminate": "Criteria for completing the drawing",
"thought_at_each_step": "Your thought process at each step",
"initialization": "Initial setup before starting the drawing"
}
PLEASE ENSURE YOUR JSON OUTPUT IS CORRECTLY FORMATTED! Avoid including extraneous words or characters outside the JSON structure.

A.1.2 Iterative reasoning

The prompt for the iterative reasoning step is as follows. Please note that current cutting-edge MLLMs cannot embed images at arbitrary positions of prompt. Therefore, we provide the image in context and refer to it in prompt description accordingly.

You are a problem visualizer, tasked with drawing an image based on a problem description using a specified tool.
You will draw the image accurately, ensuring that all information in the problem description is included in the image. This process will be done iteratively. Each iteration step will involve drawing a specific content of the figure using the tool.
The problem description is as follow: {problem}
Here is the partially completed figure given in context. {figure.png}. Please refer to it during your thought process and image updates.

//=== Continued from previous page ===
For each step, provide your thoughts in {thought_at_each_step}.
Next, update the image content {draw_content_of_each_step} using API calls with the tool tool.
Output Format:
Your output should be a JSON string. Strictly follow the JSON format provided below.
{
"thought": "Your thought on this iteration",
"is_terminal": false, // true if this is the last iteration, otherwise false. The terminition condition is {when_terminate}
"iterative_draw_step": "Description of what was drawn or modified in this step"
}
PLEASE ENSURE YOUR JSON OUTPUT IS CORRECTLY FORMATTED! Do not include any extra words or comments.

A.1.3 Conclusive reasoning

First, we apply self-alignment by prompting the MLLM to describe the content of the image.

Please describe the details of the given image accurately and comprehensively. Include all visible objects and elements without omitting any details, and avoid adding any imaginary or non-existent objects. Ensure that the description is as detailed as possible, faithfully capturing the essence of the image provided.
Here is the given image {image.png}.
Your description is as follow:

Next, the MLLM is prompted to function as a binary classifier and is tasked with checking whether the description aligns with the initial high-level plan. The prompt is shown as follow.

You are an artist who visualizes problems.
You have made a plan for your drawing: {drawing_plan}
The final content you drew is described as: {self_description}
Now, you should determine whether the description aligns with your initial drawing plan. Pay close attention to the details in both.
Your output should be a single word: 'true' if they align, and 'false' if they do not. Do not provide any additional comments or explanations.
Here is your answer:

The prompt of conclusive reasoning step is as follow.

Here is the problem: {problem}
Its visualization is given in context. {figure.png}
The description of the image is as follows: {self_description}
The trajectory of thoughts is as follows: {thoughts_trajectory}
Provide your final answer:

A.1.4 Prompt used in ablation study

In the ablation study, we investigate the impact of the planning step by removing it from the process. The removal of the planning step results in the loss of access to meta-information, such as the selected tool, which can no longer be incorporated into the iterative reasoning prompt template. To address this issue, we employ an alternative prompt template specifically designed for iterative reasoning in the absence of the planning step. This alternative template allows for the continuation of the iterative reasoning process without relying on the meta-information that would have been provided by the planning step. The specific prompt we used is shown as follow.

You are a problem visualizer, tasked with drawing an image based on a problem description using a specified tool.
You will draw the image accurately, ensuring that all information in the problem description is included in the image. This process will be done iteratively. Each iteration step will involve drawing a specific content of the figure using the tool.
The problem description is as follow: {problem}
Here is the partially completed figure given in context. {figure.png}. Please refer to it during your thought process and image updates.
For each step, provide your thoughts according to this problem.
Next, update the image content according to this problem using Python API calls.
Output Format:
Your output should be a JSON string. Strictly follow the JSON format provided below.
{
"thought": "Your thought on this iteration",
"is_terminal": false, // true if this is the last iteration, otherwise false
"iterative_draw_step": "Description of what was drawn or modified in this step"
}
PLEASE ENSURE YOUR JSON OUTPUT IS CORRECTLY FORMATTED! Do not include any extra words or comments.

A.2 Examples of Four Experimental Tasks

This section presents the input-output examples for the four tasks involved in the experiments.

A.2.1 Geometry Intersection Counting

Input: *There is a circle centered at (-1.5, -1.0) with radius 3.0. There is a polygon with coordinates [(-2.2, 4.0), (-3.2, -0.4), (2.4, -2.6), (3.8, 4.1)]. There is a line segment from (0.9, 3.3) to (-1.3, 3.4). How many intersection points are there?*

Output: *2*

A.2.2 Sudoko Puzzle

Input: *Sudoku puzzle Fill the dots with digits "[1-4]". Digits cannot repeat in the same row, column or 2x2cell. Each cell is separated by spaces or empty lines. Specify the new digits with a command: "<x> <y> <digit>", removing the quotes. The top left corner has coordinates "1 1". For example the command to add the digit "4" to the bottom left corner is "1 4 4".*

Please pay attention to the information given in the image, especially the positions of the X-axis and Y-axis.

Board:

24 31

.3 ..

.1 2.

.2 1.

Command:

Output: *1 1 4*

A.2.3 Time Series Prediction

Input: *You are a time predictor. The user will provide a sequence and you will predict the next value(only one value needed). The sequence is represented by decimal strings separated by commas.*

Please continue the following sequence without producing any additional text. Do not say anything like 'the next value in the sequence are', just return the numbers. Sequence:

458.0,387.0,427.0,565.0,465.0,445.0,450.0,556.0,500.0,452.0,435.0,554.0,510.0,433.0,453.0,548.0,
486.0,453.0,457.0,566.0,515.0,464.0,431.0,588.0,503.0,443.0,448.0,555.0,513.0,427.0,473.0,526.0,
548.0,440.0,469.0,575.0,493.0,433.0,480.0,576.0,475.0,405.0,435.0,535.0,453.0,430.0,417.0,552.0,
464.0,417.0,423.0,554.0,459.0,428.0,429.0,534.0,481.0,416.0,440.0,538.0,474.0,440.0,447.0,598.0,
467.0,439.0,446.0,567.0,485.0,441.0,429.0,599.0,464.0,424.0,436.0,574.0,443.0,410.0,420.0,532.0,
433.0,421.0,410.0,512.0,449.0,381.0,423.0,531.0,426.0,408.0,416.0,520.0,409.0,398.0,398.0,507.0,
432.0,398.0,406.0,526.0,

Output: *438.0*

A.2.4 Travelling Salesman Problem

Input: *You are an TSP solver. Solve a TSP instance:*

Here are 10 point in a city, each point is represented as a pair of numbers, indicating its coordinates in the specified space. The coordinations are as following:

0: (0.9371, 0.1482)

1: (0.6345, 0.2510)

2: (0.2628, 0.7120)

3: (0.1545, 0.9067)

4: (0.2827, 0.8081)

5: (0.9533, 0.9086)

6: (0.4199, 0.7617)

7: (0.6315, 0.0414)

8: (0.8694, 0.5878)

9: (0.7709, 0.7211)

What is the shortest possible route that visits each city exactly once and returns to the origin city?

Your answer should be the permutation of all city index (separated by comma). The start point is 0. Please give your answer directly, don't use any other tools.

Output: *0,7,1,8,9,5,2,6,4,3*

A.3 Interpretability of VAP

To better understand how VAP facilitates problem-solving, we conduct two experimental studies to gain a deeper insight.

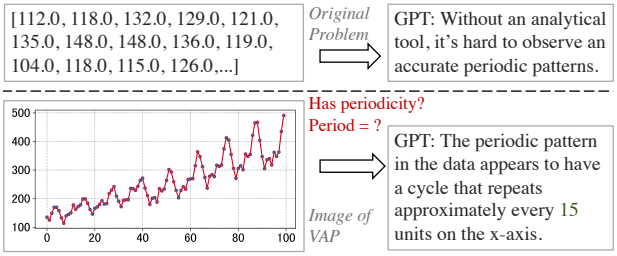
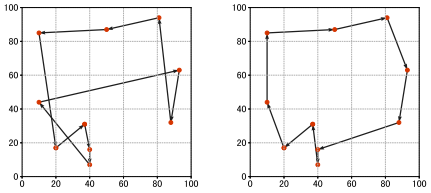


Figure 7: Detect periodicity with and without VAP.



(a) Solution given by Standard Prompting

(b) Solution given by VAP

Figure 8: A case study of TSP task.

Firstly, we focus on Time Series Prediction task. Figure 7 shows the Air Passengers Dataset from Darts. On the left, we have the textual representation of the sequence alongside an image drawn by VAP. We ask GPT-4V to model the periodicity of this sequence, which is an important factor in Time

Series Prediction. Despite the clear periodic pattern in this sequence, GPT-4V struggles to recognize this periodicity when provided only with the textual problem. In contrast, when presented with the image, GPT-4V could easily identify the sequence's periodicity. This study demonstrates how VAP can provide a more efficient and clear information format, thereby enhancing the model's reasoning ability on prediction tasks.

The second study focuses on the TSP, where we visualize two solutions for the same TSP instance in Figure 8. One solution is generated using standard prompting, while the other is given by VAP. The comparison of the two solutions reveals that the standard prompting results in numerous crossing paths, which is proven to be suboptimal in Euclidean TSP [41]. In contrast, the VAP solution has few crossing paths. This improvement is likely due to the clear visualization of visited partial paths, which provides the model with heuristic cues to easily exclude bad choices.

A.4 More Experimental Results

A.4.1 Effiency of VAP

Rregarding the computational efficiency of VAP, we conduct an experiments comparing the time consumption and accuracy of various methods across geometry and Sudoku tasks. Table 7 presents the results of this analysis.

Table 7: Efficiency and Accuracy Comparison

Method	Geometry		Sudoku	
	Time Usage	Accuracy	Time Usage	Correct Rate
Standard	0.2 s	8.5%	0.3 s	18.0%
CoT	0.5 s	10.0%	0.8 s	17.3%
CoT-SC (k=5)	2.3 s	11.0%	4.1 s	20.6%
CoT-SC (k=10)	4.5 s	11.5%	8.8 s	20.6%
ToT (n_children=5)	-	-	9.0 s	22.6%
VAP	4.1 s	**16.5%**	9.5 s	**35.5%**

Although VAP is computationally intensive, its time usage is comparable to ToT and CoT-SC (k=10) in the Sudoku task. Interestingly, for the simpler geometry task, VAP demonstrates faster performance than CoT-SC (k=10). This efficiency can be attributed to VAP's unique structure: despite its large context (including tool instructions, thought trajectory, and image encoding), the output is concise, consisting primarily of API calls and immediate thoughts. This leads to rapid inference during each step, as we observed that time usage is predominantly influenced by the number of decoded tokens. Given VAP's superior effectiveness on these tasks, we posit that the observed performance gap in computational efficiency is acceptable.

A.4.2 Different Foundational Models

Our experimental setup employs a unified VLLM. However, we observed that VAP necessitates a MLLM to process visual-text input, whereas other baselines solely require textual input. Considering that traditional LLMs are expected to outperform MLLMs in text-only tasks, we introduced GPT-4 and LLaMA 3 8B as additional LLMs to ensure a more comprehensive and equitable comparison. Table 8 presents the results of this expanded analysis on geometry intersection counting task.

Table 8: Performance on Geometry Task over Different Foundational Models

Method	Accuracy (GPT-4v)	Accuracy (GPT-4)	Accuracy (LLaMA 3)
Standard	8.5%	10.0%	7.0%
CoT	10.0%	11.0%	8.0%
CoT-SC (k = 5)	11.0%	11.5%	8.0%
CoT-SC (k = 10)	11.5%	11.5%	8.0%
CoT-SC (k = 20)	11.5%	11.5%	8.0%
VAP	**16.5%**	-	-

The findings reveal several noteworthy points. First, GPT-4 demonstrated a marginal improvement in baseline performance compared to GPT-4v, albeit still significantly lower than VAP. Conversely,

LLaMA 3 8B exhibited reduced accuracy, which can be attributed to its smaller model size potentially limiting its generalization capabilities for this particular reasoning task. While larger variants of LLaMA 3 might yield superior results, current hardware constraints precluded their evaluation. Interestingly, simpler methods such as standard prompting show greater sensitivity to model variations compared to more complex approaches like CoT with CoT-SC. Despite these differences, we believe that the fundamental conclusions of our study remain valid.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and precede the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have made main claims that accurately reflect the paper's contributions and scope in abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: The paper discuss the limitations of this work.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This paper does not include the content of the theoretical analysis results that require assumptions or proof.

   Guidelines:

   - The answer NA means that the paper does not include theoretical Results
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides all the information needed to reproduce the main experimental results.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the Results

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have provided the experimental code and test data to facilitate reproduction.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the Results See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and Baselines If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specify the all details of experiment, including hyper-parameters, the rationale for their selection and underlying model.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: In our experiment, we set the temperature of the LLM to 0 for reproducibility, ensuring that their responses were deterministic and free from randomness. And we also utilize a fixed test set. Consequently, there were no error bars in our experimental results, as the outputs were consistent and invariable across multiple runs.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: The paper does not provided resource usage in experiment because the method we proposed is LLMs-based, which solely involved API calls to these models and is not computationally intensive at all. Any computer with network communication capabilities could execute our method.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work performed has no direct societal impact, as it relies solely on calling LLMs and their behavior is fully controlled. The approach does not involve any external factors or unintended consequences that could potentially affect society.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The creators of the data and LLMs employed in this research have appropriately acknowledged and adhered to the respective licenses governing their use for research purposes.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The new assets, including source code and dataset we use, are well documented and the documentation is provided alongside the assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.