ABSEval: An Agent-based Framework for Script Evaluation

Anonymous ACL submission

Abstract

Recent research indicates that large language models (LLMs) possess a certain degree of script planning capability. However, there is still a lack of focused work on evaluating scripts generated by LLMs. The evaluation of scripts poses challenges due to their logical structure, sequential organization, adherence to commonsense constraints, and open-endedness. In this work, We introduced a novel script evaluation dataset, **MCScript**, consisting of more than 1,500 script evaluation tasks and steps, and developed an agent-based script evaluation framework, **ABSEval**, to collaboratively evaluate scripts generated by LLMs. Our experiments demonstrate that ABSEval provides superior accuracy and relevance, aligning closely with human evaluation. We evaluated the script planning capabilities of 15 mainstream LLMs and provided a detailed analysis. Furthermore, we observed phenomena like the key factor influencing the script planning ability of LLM is not parameter size and suggested improvements for evaluating open-ended questions.

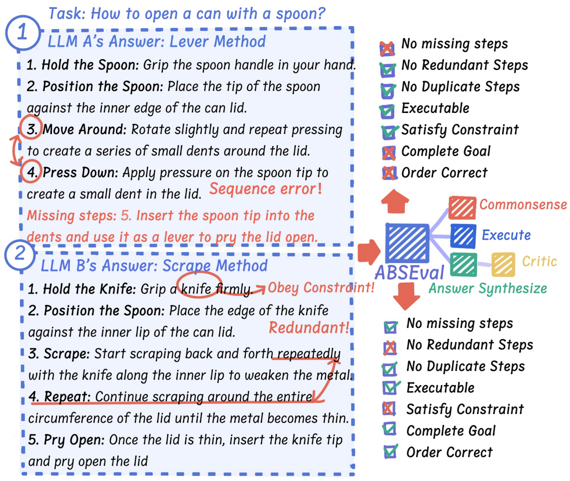

Figure 1: An example script generated to plan for "*How to open a can with a spoon?*" and evaluated using ABSEval.

1 Introduction

Script is a structure that describes an appropriate sequence of events in a particular context (Schank and Abelson, 1975; Abelson, 2014). In daily routines, individuals often rely on meticulously outlined steps to realize their objectives. For instance, Figure 1 illustrates the process of opening a can with a spoon. Recent studies have applied LLMs to script-related tasks, demonstrating that these models have scripting knowledge inside it (Sancheti and Rudinger, 2021), and they can effectively decompose high-level tasks (Huang et al., 2022). However, scripts generated by LLMs may contain errors, making it crucial to evaluate the quality of these LLM-generated scripts.

A script is a predetermined, stereotyped sequence of actions that define a well-known situation (Schank and Abelson, 1975), which is not only logically and sequentially organized but also adheres to commonsense. Additionally, entirely different steps can achieve the same goal, highlighting the open-ended nature of script tasks. Traditional approaches to script evaluation, such as manual evaluation, require considerable time and expense (Callison-Burch, 2009). Automated evaluation methods like BERTScore (Zhang et al., 2019) and Rouge (Lin, 2004) assess script correctness by calculating semantic similarity which is struggle to evaluate the sequential order of scripts. These methods require a gold answer for comparison, but it is difficult to obtain a gold answer for scripts. Furthermore, these methods have been shown to exhibit a relatively weak correlation with human judgment (Novikova et al., 2017; Chan et al., 2021).

Recent breakthroughs achieved by LLMs spurred a wave of research utilizing LLM as evaluator (Liu et al., 2023; Chiang and Lee, 2023; Zhang et al., 2023). These research underscore the LLMs' capacity to emulate human evaluative judgment closely. Even though a single LLM has demonstrated the ability to serve as an evaluator, recent research indicates that employing multiple LLMs can enhance evaluation performance (Li et al., 2023; Liang et al., 2023). Agents can be optimized for specific tasks rather than trying to encompass all capabilities within a single model. Assigning distinct roles to LLMs leads to more effectively identifying problems in text (Chan et al., 2023).

Existing script datasets are not sufficiently close to the tasks encountered in real-life scenarios, this paper introduces the **M**ulti-**C**onstrained **Script** planning dataset, i.e., **MCScript**, which includes more than 1,500 real-life script planning tasks and steps. In addition, we propose the **A**gent-**B**ased **S**cript **Eval**uation Framework (**ABSEval**), an evaluation system that integrates *Answer Synthesize Agent*, *Critic Agent*, *Execute Agent* and *Commonsense Agent* to comprehensively evaluate the scripts based on their different characteristics. We designed an *Answer Synthesize Agent* to act as a learner, learn scripts generated by LLMs being evaluated, and produce a more refined answer. Then, a *Critic Agent* compares the scripts under evaluation with the gold answer provided by the *Answer Synthesize Agent*, identifying mistakes such as missing, redundant, and duplicate steps. Moreover, an *Execute Agent* verifies whether the scripts meet the implicit constraints of tasks, achieve the desired goals, and maintain a logical sequence by executing each step of the scripts. Finally, a *Commonsense Agent* assesses whether each step of the script conforms to commonsense.

This paper evaluated 15 widely used LLMs and analyzed their script planning capabilities. From the evaluation results, we observed some interesting phenomena, like the fact that the key factor influencing the script planning ability of LLM is not the LLM's parameters, providing gold answers within appropriate metrics can improve the assessment performance of open-ended questions, etc.

Our contributions are as follows: *1)* We developed a high-quality script evaluation dataset **MCScript**, which simulates a real-world situation by adding multiple constraints and contains over 1,500 script tasks and answers. *2)* We propose **ABSEval**, an agent-based evaluation framework that exhibits superior alignment with human evaluations compared to current script assessment methods. *3)* Using **ABSEval**, we assessed the script planning capabilities of 15 LLMs, offering insights into the advancements in LLMs' script planning abilities.

2 Data Construction

Currently, multiple large-scale script datasets are developed via crowdsourcing or automatic methods(Wanzare et al., 2016; Regneri et al., 2010; Lyu et al., 2021). However, these datasets concentrate on abstract questions (e.g., *Create a decision tree.*). We aim to create a set of evaluation data that is more closely aligned with real-life specific tasks (*e.g., Create a decision tree on computer to help you choose a holiday destination.*). We utilized WikiHow (Koupaee and Wang, 2018), a comprehensive database of how-to guides on a wide range of subjects, as the primary source for our data. From this resource, we selected abstract questions across ten different topics. As is shown in Table 1, we adopt the in-context learning (Brown et al., 2020; Work) for GPT-4-turbo[1] to expand the initial set of abstract questions by adding one to three constraints to each, thereby enhancing their relevance and realism. After the expansion, a thorough review of the newly formulated questions was conducted to select high-quality evaluation questions. The specific number of questions and types of topics are detailed in Figure 2.

Prompt: Create possible specific goals according to the abstract goal, here is an example. **Abstract task:** Create a decision tree **Constraint:** on computer, to help you choose a holiday destination, with 3 options **Constraint task:** Create a decision tree on computer to help you choose a holiday destination with 3 options.
Obtain abstract task: How to buy Disney World tickets **Add constraints:** Online, For a family of four, During peak season. **Generate constraint question:** Research and purchase Disney World tickets online for a family of four during peak season.

Table 1: An example of prompt for generating a constraint script task.

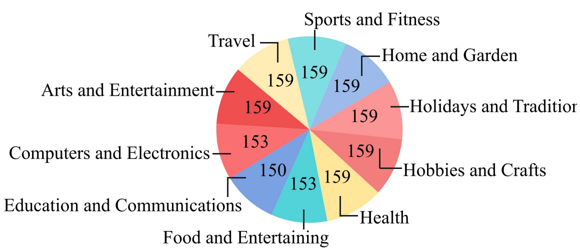

Figure 2: Distribution of topic categories and the number of tasks in MCScript.

[1] https://openai.com/index/gpt-4/

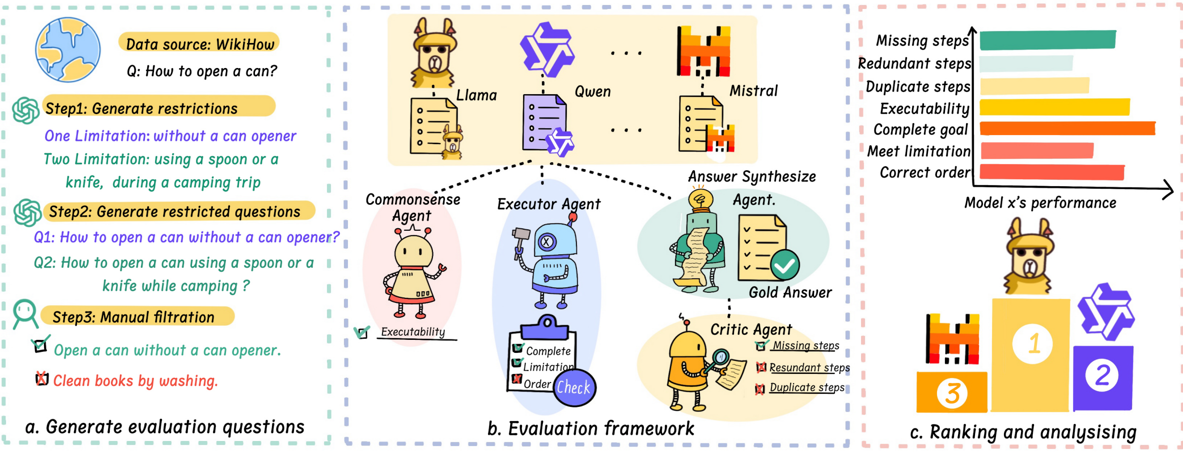

Figure 3: The evaluation process of LLM using ABSEval. We first obtained abstract problems from wikiHow and used GPT-4-turbo to add constraints, followed by manual screening to select high-quality questions. Subsequently, we utilized the ABSEval framework to complete the evaluation process. Finally, we analyzed the models' script planning capabilities based on the evaluation results.

3 Evaluation Methodology

This section provides an in-depth explanation of the ABSEval evaluation framework. The discussion includes a breakdown of the evaluation metrics, the components in the evaluation framework, and a detailed explanation of the entire evaluation process. The overall workflow is illustrated in Figure 3.

3.1 Evaluation Metrics

As we stated in Section 1, the logical structure, sequential nature, and adherence to the commonsense of scripts present challenges for their evaluation. Evaluating such scripts necessitates methodologies distinct from those applied to traditional text generation. To address these distinctive script features, we devised specialized evaluation criteria.

Our evaluation metrics focus on three key aspects. Firstly, we introduced three evaluation criteria to assess the completeness and correctness of the logical structure: (1) *No Missing Steps*: ensuring all critical steps are included. (2) *No Redundant Steps*: the script contains no unnecessary steps. (3) *No Duplicate Steps*: avoiding repetition of actions. Secondly, to evaluate the script's adherence to commonsense knowledge, we introduced (4) *Executable*: ensuring alignment with common sense knowledge. Finally, to check the sequential order of the script and whether it achieves its goal, we defined the criteria: (5) *Satisfy Constraint*: meeting implicit task constraints. (6) *Complete Goal*: achieving the intended objective. (7)*Order Correct*: maintaining a logical sequence of steps.

3.2 ABSEval Framework

Considering the limitations of script evaluation by a single LLM, our study embraces an agent-based paradigm for our evaluation framework. We demonstrated that collaborative effort affords a more human-aligned assessment than a single LLM in Section 4. By comparing different LLMs against human-annotated standards, we opted for Qwen-110B-Chat[2] to serve as the evaluation backbone within our ABSEval framework. Our study concentrates on the deployment of homogeneous sets of LLMs, meaning all agents are represented by the same LLM. The prompt for each agent is detailed in Appendix A.1.

Answer Synthesize Agent. Due to the diversity and open-ended nature of scripts, there is no standard answer for reference. It is challenging to directly identify errors within them. To address this, we employed a pooling strategy where the *Answer Synthesize Agent* learns from the scripts to be evaluated for the same task and synthesizes an enhanced gold answer. By comparing the scripts to this gold answer, it becomes easier to identify implicit errors.

Critic Agent. Once the *Answer Synthesize Agent* has crafted the gold answer, the *Critic Agent* checks the scripts up for evaluation against this gold answer to identify errors. We demonstrated that these errors tend to be subtle, they can be better identified

[2]We compared GPT-4-turbo, GPT-3.5-turbo, and Qwen-110B-Chat, ultimately selecting Qwen-110B-Chat (`https://huggingface.co/Qwen`) for its closest alignment with human evaluations.

by comparing them with gold answers generated by *Answer Synthesize Agent* in Section 4.2. Through the collaboration of the *Answer Synthesize Agent* and the *Critic Agent*, we can identify missing steps, redundant steps, and duplicate step errors within the scripts.

Execute Agent. To confirm whether a script successfully attains its intended objective without logical or sequential errors, we delegate the role of the executor to an LLM. We first guide the Execute Agent to execute the script to be evaluated step-by-step by providing the prompt "*I have provided you with the steps to complete the task:[SCRIPT]. Please follow these steps and answer my questions below...*". Then assesses whether the final goal has been achieved, whether the implicit constraints of the task have been satisfied, and whether there are any errors in the sequence of steps.

Commonsense Agent. Scripts generated by LLMs occasionally include steps at odds with commonsense reasoning (*e.g., Washing the book with water to achieve the purpose of cleaning.*). Hence, we incorporate a Commonsense Agent. Its task is to ascertain the concordance of scripted steps with commonsense knowledge. We employ an LLM as our Commonsense Agent to identify if there were parts of the script steps that did not follow commonsense.

4 Experiments

4.1 Evaluated Models

Our primary focus for evaluation lies in open-source models with parameter ranging from 6 billion to 70 billion, including LLaMa2-7b-Chat (Touvron et al., 2023), LLaMa2-13b-Chat, LLaMa2-70b-Chat, LLaMa3-8b-Instruct, LLaMa3-70b-Instruct, Baichuan-13B-Chat (Yang et al., 2023), Baichuan2-13B-Chat, Qwen-7B-Chat(Bai et al., 2023), Qwen-14B-Chat, Qwen-72B-Chat, Mistral-7B-Instruct-v0.2(Jiang et al., 2023), Mistral-7B-Instruct-v0.1, Mistral-8x7B-Instruct-v0.1, Vicuna-7b-v1.5, Vicuna-13b-v1.5. We added the prompt *"Let's think step by step"* to guide the models in generating script responses, which is a simple strategy to enhance the reasoning performance of the models (Kojima et al., 2022).

4.2 Results

Can ABSEval better align with human evaluations? To prove that the proposed ABSEval could be closer to human evaluations compared with the previous evaluation approaches, we randomly selected 200 scripts generated by LLMs for manual annotation. Subsequently, we tested three state-of-the-art LLMs, GPT-3.5-turbo (Ouyang et al., 2022), GPT-4-turbo (Achiam et al., 2023), and Qwen-110B-Chat, for the ABSEval assessment. Additionally, we queried a single LLM directly to evaluate the seven metrics in ABSEval based on the same scripts for comparison. A better evaluation would obtain results similar to those obtained by human annotations.

The Mean Squared Error (MSE) values for the seven metrics of ABSEval and Single-LLM against human evaluations were calculated. As shown in Table 2, Qwen-110B-Chat excelled in performance in both the ABSEval and Single-LLM frameworks. A single-LLM evaluation system, while incorporating advanced models, may fall short of providing a comprehensive analysis that matches human evaluators' results effectively. In contrast, the ABSEval evaluation system significantly enhances the alignment of LLM assessments with human judgment.

LLM	Mechanism	MSE
Qwen-110-Chat	ABSEval	**0.087**
GPT-4-turbo	ABSEval	0.174
GPT-3.5-turbo	ABSEval	0.329
Qwen-110-Chat	Single-LLM	**0.257**
GPT-4-turbo	Single-LLM	0.29
GPT-3.5-turbo	Single-LLM	0.361

Table 2: Similarity of evaluation results to human assessments for GPT-3.5-turbo, GPT-4-turbo, and Qwen-110B-Chat as LLMs in ABSEval and Single-LLM.

Should Gold answers be provided for evaluating the open-end questions? To answer this question, we investigate the potential advantages of including a gold answer when assessing open-ended questions like scripts for the automatic evaluation. Our analysis of the data presented in Figure 4 involved comparing the coherence between the evaluation of Qwen-110B-Chat and human evaluation across various metrics, both with and without a gold answer. The findings of our study indicate that incorporating a gold answer can assist the model in identifying missing steps more effectively. However, it was observed that the presence of a gold answer can also reduce the accuracy of the model's assessments concerning step sequencing correctness, goal achievement, and adherence to implicit constraints. Providing a reference answer

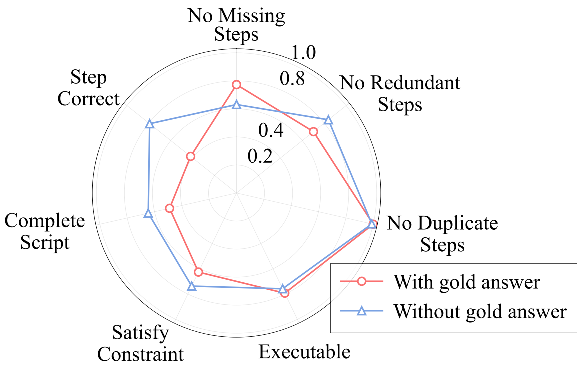

Figure 4: Comparing the consistency of evaluation results with human assessments when directly using LLM for evaluation, with and without providing an answer.

can assist in evaluating some metrics but may also lead to performance degradation for some evaluation metrics. Therefore, it is crucial to establish an appropriate evaluation method, such as ABSEval, to provide gold answers for certain evaluation metrics.

Can Answer Synthesize Agent generate high-quality answers? We utilized the answers generated by the *Answer Synthesize Agent* and Qwen-110B-Chat as the gold answers for the Critic Agent to evaluate. We then compared the consistency of both evaluation results of *Critic Agent* with human-labeled data. Table 3 demonstrates the performance differences, showing that the *Answer Synthesize Agent* outperforms the direct answers from Qwen-110B-Chat on all three metrics of *No Missing Steps, No Redundant Steps, and No Duplicate Steps*.

Gold answer generation	NM	NR	ND
Answer Synthesize	0.895	0.965	1.0
Qwen-110B-Chat	0.75	0.855	1.0

Table 3: Comparison of the different gold answer generation approaches. NM: No Missing Steps, NR: No Redundant Steps, ND: No Duplicate Steps.

Can ABSEval effectively identify errors in scripts? To answer this question, we introduced some perturbations to the completely correct script and evaluated it using the ABSEval framework. We used GPT-4-turbo to introduce perturbations into completely correct script steps (*e.g., Remove a key step in the script*), and the perturbations construction prompt is detailed in Table 9. For each evaluation metric in ABSEval, we constructed 50 perturbation scripts and then used ABSEval to evaluate them. We calculated the Accuracy(Acc.) of ABSEval in identifying each type of interference error, as shown in Table 4, ABSEval effectively identified all types of errors, demonstrating the validity of the ABSEval framework.

Perturbations category	Acc.
Missing steps	0.84
Redundant steps	0.96
Duplicate steps	0.96
Satisfy Constraint	0.85
Complete	0.92
Step order	0.84

Table 4: Accuracy of ABSEval checking perturbations errors.

4.3 Evaluating Scripts in different LLMs by ABSEval

The overall evaluation results of ABSEval are shown in Table 5.

What are the most common errors in all LLMs during script planning?

We categorized the LLMs in Table 5 based on their parameter sizes, and plotted a heat map about the overall performance of different parameter levels in Figure 5. As shown in Figure 5, the most frequent issues encountered in LLMs during script planning involve missing steps and failing to achieve the intended goal. In contrast, the problems of redundant steps appear to be relatively uncommon. An increase in the model's parameter size correlates with improved accuracy across various metrics. Despite this, even LLMs with up to 70 billion parameters struggle to perform well across all metrics.

How do LLMs perform across different script planning topics?

The heat map in Figure 8 in the appendix shows that LLMs perform best on topics related to *Education and Communications*, while their weakest performance is on topics related to *Health*. Notably, the heatmap uncovers substantial performance variations across different topics. We believe that the existence of this difference is related to the knowledge stored within the LLMs.

Model Name	Rank	Size	NM	NR	ND	EX	SC	CG	OC
Baichuan-Chat	14th	13B	0.029	0.787	0.994	0.833	0.673	0.572	0.632
Baichuan2-Chat	13th	13B	0.139	0.777	0.992	0.813	0.677	0.580	0.604
Vicuna-v1.5	10th	7B	0.044	0.811	0.995	0.876	0.713	0.611	0.696
Vicuna-v1.5	9th	13B	0.074	0.858	0.999	0.888	0.708	0.624	0.720
LLaMa2-chat	11th	7B	0.250	0.728	0.999	0.836	0.661	0.566	0.709
LLaMa2-chat	7th	13B	0.211	0.807	0.999	0.871	0.715	0.622	0.722
LLaMa2-chat	2nd	70B	**0.379**	0.773	0.999	0.886	0.711	0.665	0.727
LLaMa3-instruct	5th	8B	0.103	0.880	**1.000**	0.889	0.758	0.681	0.725
LLaMa3-instruct	1st	70B	0.154	0.894	1.000	0.902	0.755	**0.711**	0.745
Mistral-Instruct-v0.1	15th	7B	0.048	0.703	0.998	0.816	0.671	0.565	0.610
Mistral-Instruct-v0.2	6th	7B	0.220	0.810	1.000	0.889	0.713	0.666	0.718
Mistral-8x7B-Instruct-v0.1	4th	8x7B	0.092	0.888	**0.999**	0.902	0.753	0.685	**0.766**
Qwen-Chat	12th	7B	0.089	0.831	0.996	0.862	0.678	0.564	0.668
Qwen-Chat	8th	14B	0.139	0.878	0.997	0.879	0.719	0.593	0.703
Qwen-Chat	3rd	72B	0.129	**0.913**	0.998	0.900	**0.763**	0.654	0.763
ALL	-	-	0.137	0.824	0.998	0.870	0.712	0.624	0.700

Table 5: The accuracy rate of all evaluation LLMs for different metrics on the MCScript data set. NM: No Missing Steps, NR: No Redundant Steps, ND: No Duplicate Steps, EX: Executable, SC: Satisfy Constraint, CG: Complete Goal, OC: Order Correct.

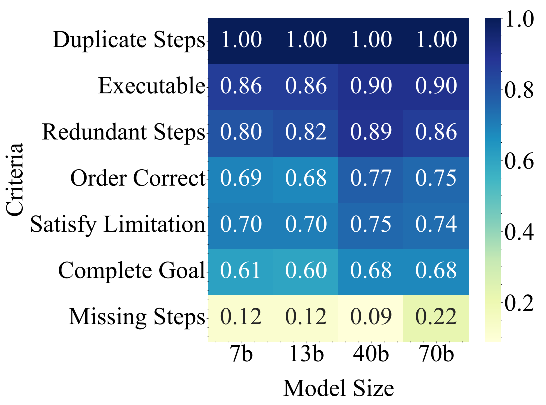

Figure 5: The heat map depicts the relation of model size and evaluation criteria.

5 Deepthinking ABSEval

We present the performance of all LLMs be evaluated across various metrics in Figure 10 in the appendix. To enhance the clarity of our observations, we employ a consistent color scheme to delineate LLMs within the same series (*e.g., LLaMa3 is shown in red*), with varying shades denoting differences in LLM parameters. Our analysis has several interesting observations.

Distinct LLM series employ domain-specific strengths.

In our comparative analysis, no single LLM demonstrated superiority across every evaluation metric. For instance, both the LLaMa2 and LLaMa3 models exhibit prowess in reducing missing steps, ensuring adherence to constraints, and effectively realizing intended goals. Meanwhile, Qwen displays a remarkable ability to reduce redundant actions, demonstrating heightened efficiency in certain problem-solving scenarios. The Vicuna model's strength lies in its strong compliance with commonsense constraints. Overall, different models have advantages in different evaluation metrics. These findings underscore the potential for future enhancements in the domain-specific proficiencies of LLMs.

Larger parameter size does not necessarily guarantee superior metric performance.

As shown in Figure 5, a larger number of model parameters generally leads to improved performance in script planning tasks. More parameters are associated with fewer missing steps, improved goal accomplishment, and better sequence maintenance. However, this trend is not consistent across all cri-

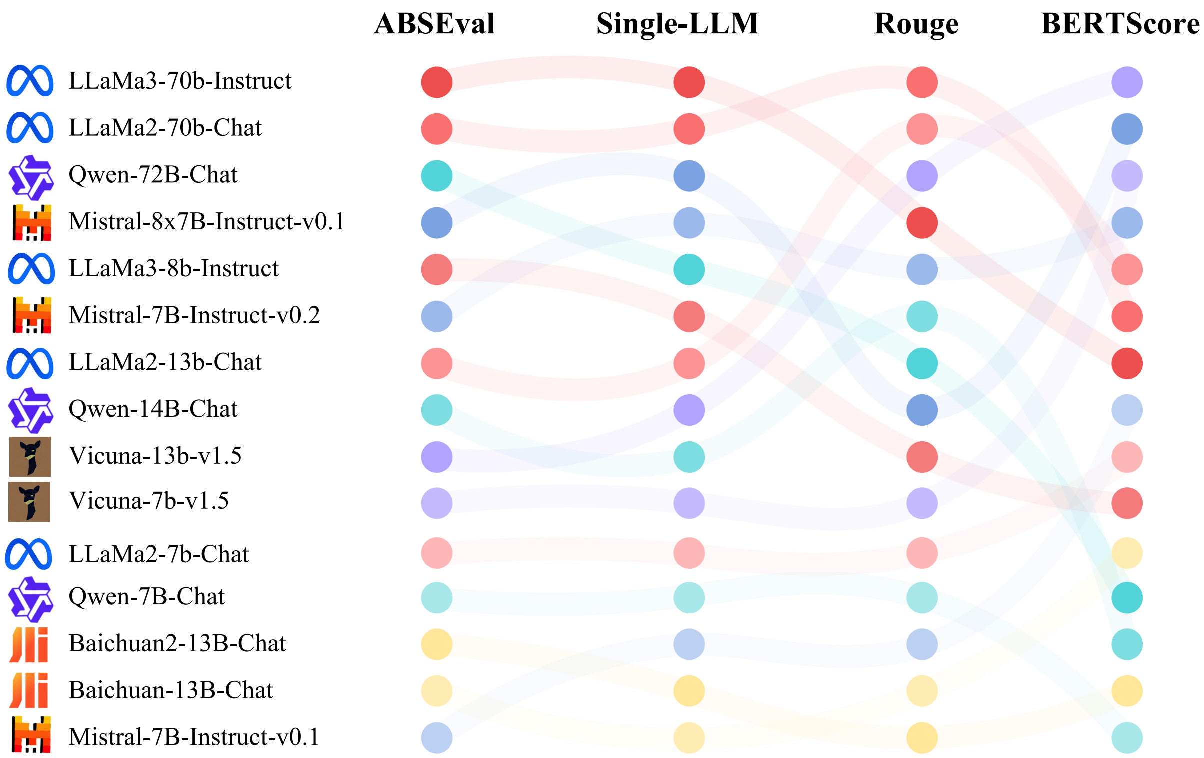

Figure 6: Comparison of different evaluation metrics, including out ABSEval, Sigle-LLM evaluation, Rouge, and BERTScore.

teria. Notably, within the LLaMa2 series, a higher parameter count led to an increase in redundant steps, contrary to expectations. This decline in performance with increased parameters may be linked to decreased efficiency in following instructions, resulting in responses that include content beyond the task requirements.

Factors beyond parameter size impact LLMs' script planning capabilities.

While some metrics show improved outcomes with larger parameters, models within the same series maintain a consistent rank order across different metrics. For instance, the LLaMa2 and LLaMa3 series generally outperform the Qwen series in the *'No Missing Steps'* metric. Remarkably, the Qwen-72B-Chat model, with 70 billion parameters, did not outperform the LLaMa2 and LLaMa3 series models in this metric, despite its significantly larger parameter count. Additionally, in the *'No Redundant Steps'* metric, the Qwen and LLaMa3 series models often outperformed the LLaMa2 series models. Even the LLaMa2-70B-Chat model failed to surpass the Qwen-7B-Chat. We believe that diverse training conditions such as pre-training data, architecture, and methodologies unique to each model series play a crucial role in determining script planning proficiency. Thus, factors beyond mere parameter size play a significant role in enhancing the script planning capabilities of LLMs.

LLMs perform better on tasks with more steps.

We analyzed the relationship between LLMs' performance on four metrics (*Correct Order, Executable, No Redundant Steps, and Satisfy Constraints.*) and the number of steps in the script. As illustrated in Figure 11 in the appendix, we observed that as the steps of script tasks increased, LLMs exhibited improved accuracy in maintaining logical sequences and adhering to constraints. Furthermore, as the steps of script tasks increased, the occurrence of redundant steps decreased. This trend may arise from LLMs' tendency to focus on crucial steps and avoid unnecessary redundancy when addressing complex issues. Overall, LLMs demonstrate better performance on more step script tasks, indicating their existing capability in handling complex planning tasks.

Limitations of current script evaluation methods

A sample of 1,000 questions from MCScript was randomly selected for critical analysis of the limita-

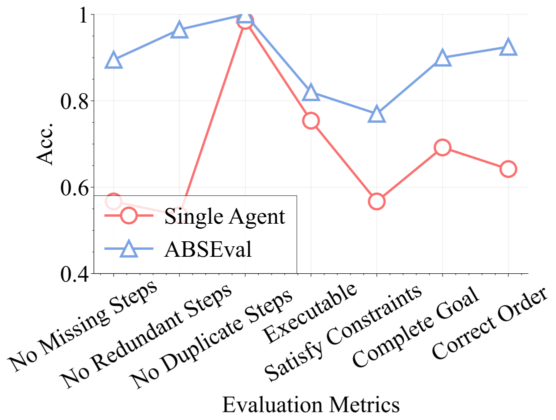

Figure 7: The consistency of Single-Agent and ABSEval with manual evaluation in each metric.

tions of different evaluation methods. The evaluation was conducted on 15,000 scripts generated by 15 different LLMs using ABSEval, Single-LLM, ROUGE, and BERTScore. The comparison of the rankings generated by each method can be seen in Figure 6.

In contrast to traditional methods such as BERTScore (Zhang et al., 2019) and ROUGE (Lin, 2004), our evaluation approach presents several advantages. The open-ended nature of scripts allows for a variety of sequences to achieve the same goal. BERTScore evaluates text by comparing the cosine similarity of each embedding vector in the generated text with the reference text, while ROUGE assesses similarity based on the longest common subsequence between the two texts. These methods heavily depend on the reference answer, leading to significant inaccuracies when evaluating scripts that differ greatly from the reference but still meet the objective. Additionally, these methods struggle to assess the sequential flow of script steps and logical structure. Therefore, traditional evaluation methods do not offer a fair and comprehensive evaluation of scripts, resulting in varying LLM performance rankings compared to our evaluation method.

As discussed in Section 4, ABSEval more closely aligns with human preferences compared to Single-LLM. Figure 7 highlights the comparison of ABSEval and Single-Agent in terms of consistency with human annotations across various evaluation metrics. Notably, Single-Agent performed poorly in categories such as *No Missing Steps*, *No Redundant Steps*, and *Satisfy Constraints*, which demonstrates that distributing detail tasks in agents can effectively optimize evaluation performance.

6 Related Work

Scripts A structure describing a sequence of events in a particular scenario is script (Schank and Abelson, 1975). The current work is focused on extracting script knowledge from LLMs. For instance, Lyu et al. (2021) introduced a model that generates a series of steps designed to achieve a specified objective. Huang et al. (2022) showed that LLMs can effectively break down high-level tasks into mid-level plans even without additional training. Yuan et al. (2023) proposed a method to enhance LLMs by first over-generating and then filtering their output, thereby refining script generation when multiple constraints are in play. The emphasis of these advancements has largely been on improving the generative aspects of models. There is a notable scarcity of research on establishing comprehensive and fair evaluation methods for evaluating the script planning abilities of LLMs.

Open-ended Text Evaluation Evaluating open-ended text poses significant challenges due to the intensive nature of human-based methods. Traditional metrics like BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) often diverge from human judgments. The capabilities of LLMs offer new thinking for text assessment. For instance, G-EVAL (Liu et al., 2023) employs LLMs with chain-of-thought processes and a form-filling approach to evaluate NLG outputs. Advances with collaborative LLMs show promise in aligning more closely with human ratings. Mandi et al. (2023) introduced a method for multi-robot collaboration for both strategic communication and detailed path planning, Chan et al. (2023) created an agent-based debate framework for text evaluation.

7 Conclusion

In this study, we introduced a new script evaluation dataset, **MCScript**, comprising over 1,500 script tasks and steps. We proposed a more fair, fine-grained, and human-aligned script evaluation method known as **ABSEval**. By utilizing ABSEval, we conducted a comprehensive analysis of the script planning abilities of 15 current LLMs and identified the shortcomings of existing script evaluation methods. Our discussion and analysis provide insights for the evaluation of open-ended tasks similar to scripts. Our objective is to establish a new framework within the LLM community for assessing and analyzing the script planning capabilities of LLMs.

8 Limitation

In our proposed ABSEval, we use homogeneous LLMs, meaning all roles are performed by the same LLM. Future work could explore using heterogeneous LLMs, assigning tasks based on the strengths of different LLMs to further enhance the potential of the evaluation framework. Additionally, Our dataset still contains a small number of errors because the data volume is too large for manual checking, which is overly time-consuming. Last but not least, all our evaluation metrics are binary (*True or False*). It can further optimize the evaluation granularity by assessing the degree of completion for each metric (e.g., *how many steps are missing*, *how many constraints are not met*, etc.).

9 References

References

Robert P Abelson. 2014. Script processing in attitude formation and decision making. In *Cognition and social behavior*, pages 33–45. Psychology Press.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Chris Callison-Burch. 2009. Fast, cheap, and creative: Evaluating translation quality using amazon's mechanical turk. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 286–295.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*.

CS Richard Chan, Charuta Pethe, and Steven Skiena. 2021. Natural language processing versus rule-based text analysis: Comparing bert score and readability indices to predict crowdfunding outcomes. *Journal of Business Venturing Insights*, 16:e00276.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? *arXiv preprint arXiv:2305.01937*.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large scale language model society.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Qing Lyu, Li Zhang, and Chris Callison-Burch. 2021. Goal-oriented script construction. *arXiv preprint arXiv:2107.13189*.

Zhao Mandi, Shreeya Jain, and Shuran Song. 2023. Roco: Dialectic multi-robot collaboration with large language models. *arXiv preprint arXiv:2307.04738*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 979–988.

Abhilasha Sancheti and Rachel Rudinger. 2021. What do large language models learn about scripts? *arXiv preprint arXiv:2112.13834*.

Roger C Schank and Robert P Abelson. 1975. Scripts, plans, and knowledge. In *IJCAI*, volume 75, pages 151–157. New York.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Lilian DA Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. A crowdsourced database of event sequence descriptions for the acquisition of high-quality script knowledge.

What Makes In-Context Learning Work. Rethinking the role of demonstrations: What makes in-context learning work?

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Siyu Yuan, Jiangjie Chen, Ziquan Fu, Xuyang Ge, Soham Shah, Charles Robert Jankowski, Yanghua Xiao, and Deqing Yang. 2023. Distilling script knowledge from large language models for constrained language planning. *arXiv preprint arXiv:2305.05252*.

Baoli Zhang, Haining Xie, Pengfan Du, Junhao Chen, Pengfei Cao, Yubo Chen, Shengping Liu, Kang Liu, and Jun Zhao. 2023. Zhujiu: A multi-dimensional, multi-faceted chinese benchmark for large language models. *arXiv preprint arXiv:2308.14353*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Appendices

A.1 Prompt Format

The detailed prompt of construct MCScript is shown in Table 6 and the prompt of each agent in ABSEval is shown in Table 7.

A.2 Eval Examples

We provide a specific example evaluated using the ABSEval framework in Table 8.

A.3 Experiment Details

Table 9 illustrates the specific prompts with added perturbations. To demonstrate the validity of the ABSEval framework, we removed the Answer Synthesize Agent and queried Qwen-110B-Chat to directly generate the standard answers. We compared the consistency of the *Critic Agent*'s evaluation results with human annotations between the two methods. As shown in Table 3, generating the gold answer through the Answer Synthesize Agent significantly improves the accuracy of the *Critic Agent*'s judgments.

A.4 The performance of the model in ABSEval

Topic heat map. Figure 8 presents a heatmap of the performance of the LLMs participating in the evaluation across all topics.

Overall performance of metrics. Figure 9 shows the overall performance of all LLMs across the seven evaluation metrics in ABSEval.

Detailed analysis of each metrics. Figure 10 analyzes the performance of each participating LLM for each metric. Models from the same series are drawn in the same color, with darker shades representing larger parameter sizes.

The relationship between LLM performance and script length. Figure 11 illustrates the relationship between model performance and the length of script tasks across four different metrics.

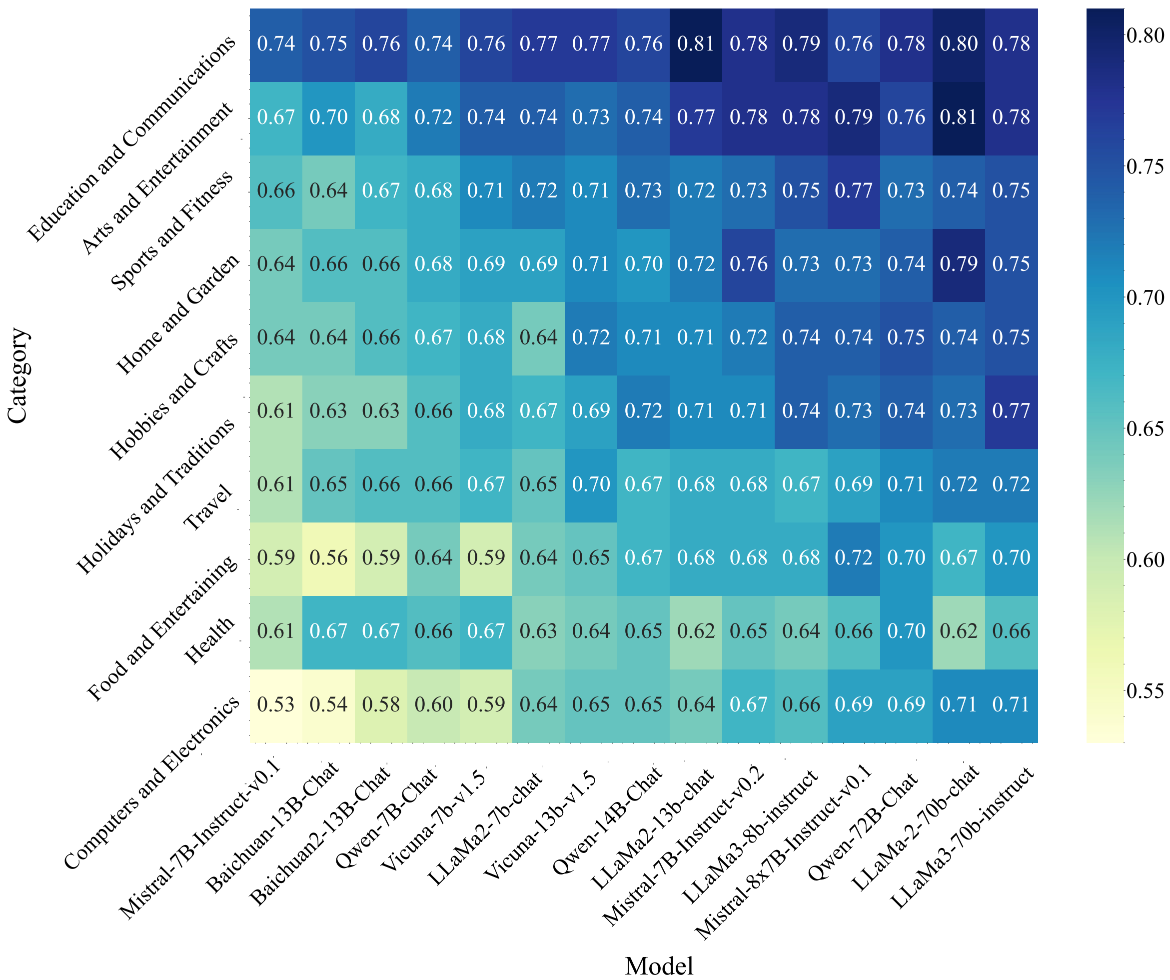

Figure 8: The heat map of all LLMs in different question topics.

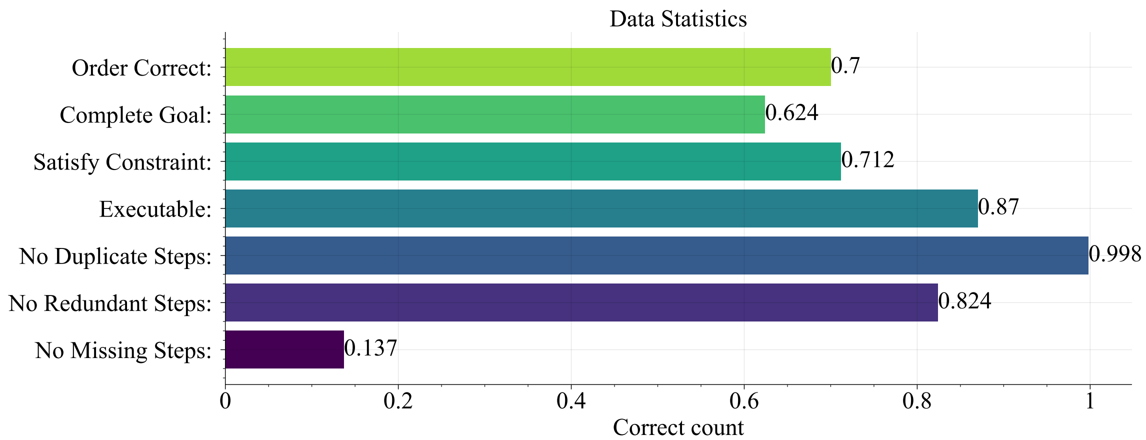

Figure 9: The accuracy of all LLMs in the metric of ABSEval.

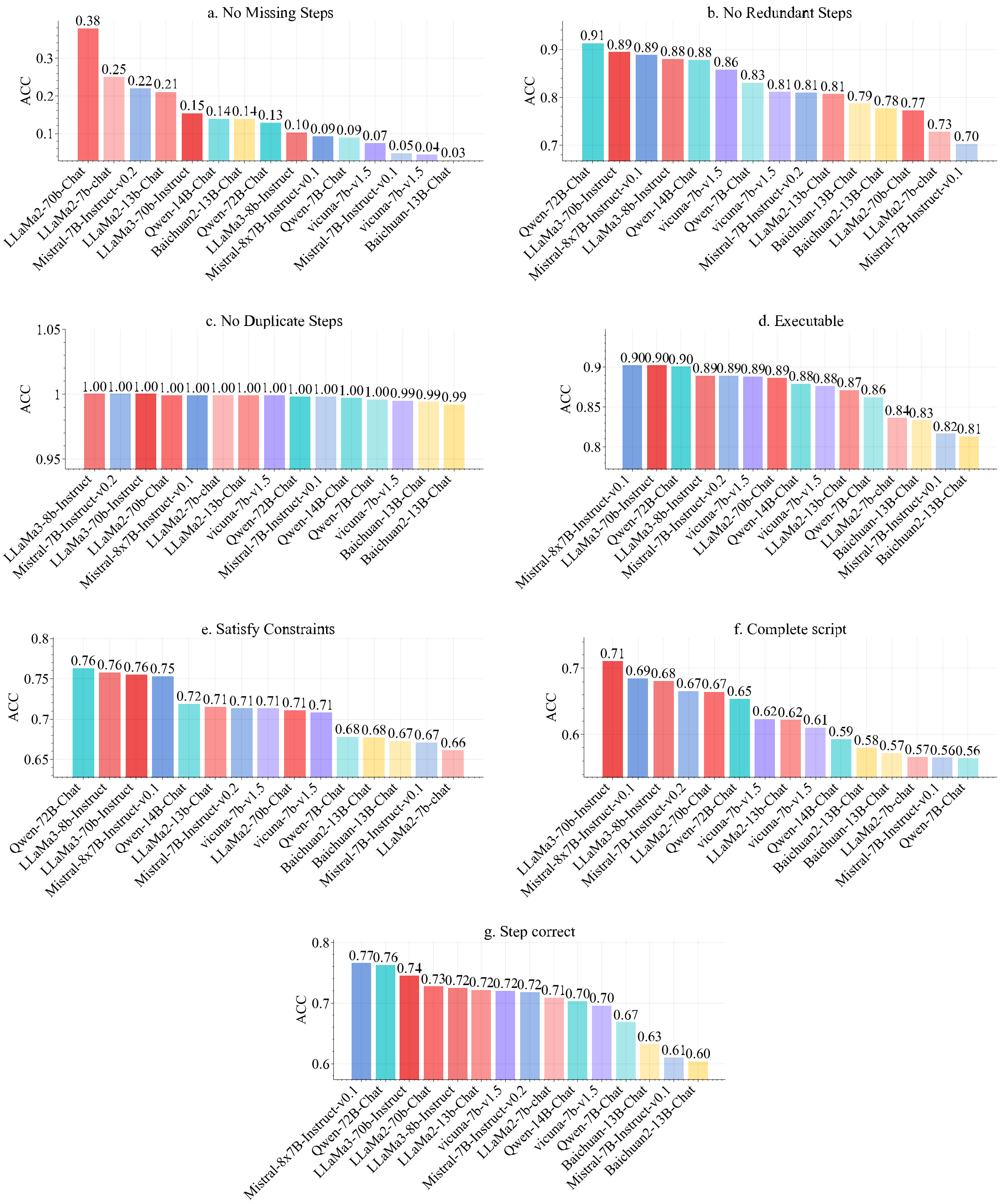

Figure 10: Performance of all LLMs in ABSEval metrics

Step1: Obtain abstract question
Source: WikiHow **Question:** How to buy Disney World tickets
Step2: Add constraint and generate questions with constraints
Prompt: Create possible specific goals according to the abstract Goal, here are some examples Abstract Goal: Create a Decision Tree { "Constraint": "on Computer", "Specific Goal": "Create a Decision Tree on a Computer" } Here is my question: Abstract Goal: {ABSTRACT QUESTION} Please answer me in JSON format {"Constraint": "...", "Specific Goal": "..."}.
One constraint: Online **Two constraints:** Online, For a family of four **Three constraints:** Online, For a family of four, During peak season
Question one: Learn how to buy Disney World tickets online **Question Two:** Research how to buy Disney World tickets online for a family of four **Question Three:** Research and purchase Disney World tickets online for a family of four during peak season.

Table 6: An example of generating a restricted script task.

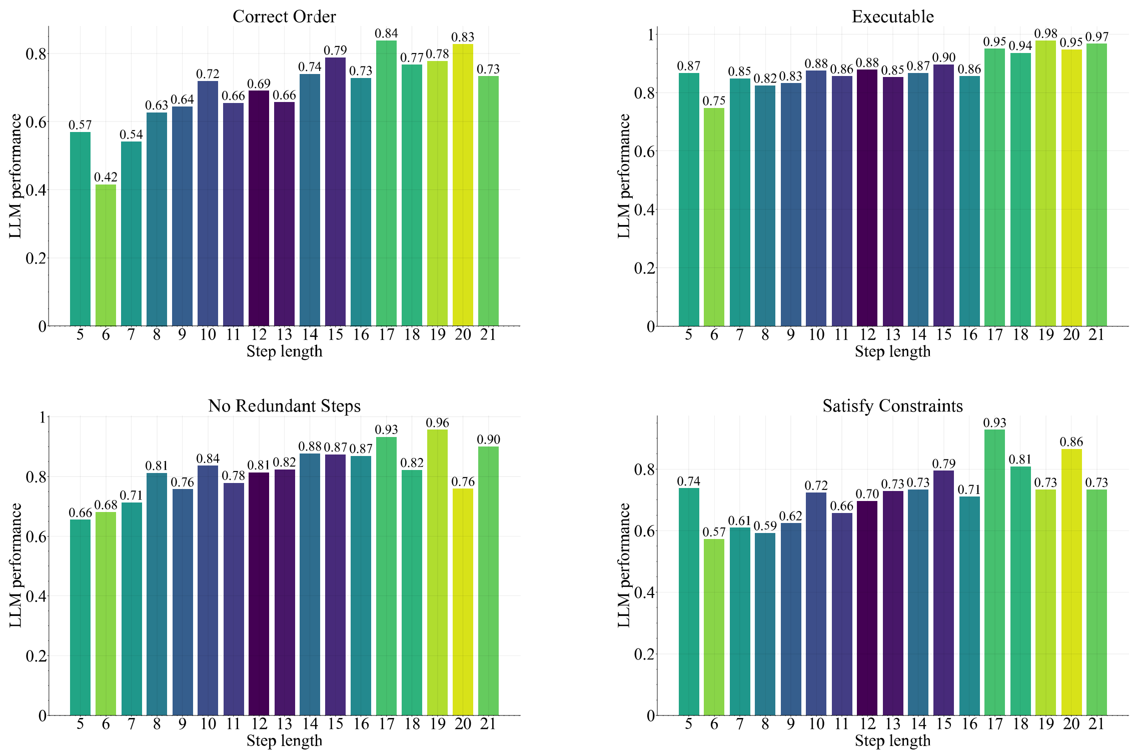

Figure 11: Performance of all LLMs about response length.

Answer Synthesize Agent
Now I want you to play the role of a learner, I hope you can help me complete this planning task through your own knowledge and learning from other examples.
The task is: [Tasks]
Here are some examples, but note that these examples may have flaws. I hope you can provide me with comprehensive guidance based on these examples. [EXAMPLES]
If you do not think these examples are useful, you can give your answer directly.
Please pay attention! Answer me in the following format and ensure that each step is concise: 1...,2...,3...,... Do not answer irrelevant content.

Ctitic Agent
Please play the role of an evaluator, the question that needs your evaluation is [Tasks].
The standard answer is: [Gold Answer]
The answer I need your evaluation is:[Model Answer]
I would like you to check if there are any missing, redundant, or duplicated steps in these steps.
missing steps: The script is missing any steps.
redundant steps: There are steps unrelated to achieving the goal.
duplicate steps: There are duplicate steps present.
Let's think step by step.
Please answer me in the JSON format:
{ "missing_steps": "True",
"redundant_steps": "True",
"duplicate_steps": "True",
"explain": "This script is missing key step XXX. Step x is not related to the target and belongs to redundant steps, but there are no duplicate steps..." }

Execute Agent
Now please play the role of an executor to complete this task: [QUESTION]. There are Constraints to the task: [CONSTRAINT]. I have provided you with the steps to complete the task:[MODEL INFERENCE]. Please follow these steps and answer my questions below
1. The script meets the constraints [CONSTRAINT], meet_constraints is True. The script doesn't meet the constraints [CONSTRAINT], meet_constraint is False.
2. The script achieves the goal, and the complete goal is True. If the script doesn't achieve the goal, the complete_goal is False.
3. The sequence of scripts is correct, and step_order_correct is True. The sequence of scripts is wrong, and the step_order_correct is False.
Let's think step by step.
Please answer me in the JSON format:
{ "meet_constraint": "False",
"complete_goal": "True",
"step_order_correct": "False"
"explain": "This script did not meet the constraints given in the question. In addition, there is an error in the sequence of steps, because ..." }

Commonsense Agent
Please play the role of an evaluator.
Please evaluate this answer: [MODEL INFERENCE]
Please determine if any areas in this script do not conform to commonsense.
If each step of this script conforms to common sense, please answer me True, otherwise, answer me False. Please answer me only with True or False.

Table 7: The detailed prompt of each agent in ABSEval

Question: Develop an Online Calculator to Determine the Correct Amount of Baking Soda to Drink for Health Benefits Based on Individual Body Weight and Age

Constraints: For Health Benefits, Using Measured Amounts

Mistral-7B-Instruct-v0.2 Answer:
1. Dissolve 2-3 teaspoons of baking soda in a glass of lukewarm water (about 4 ml).
2. Stir the mixture until the baking soda is completely dissolved.
3. Drink the baking soda water slowly.
4. Consume the solution on an empty stomach for optimal effectiveness.
5. Follow up with a full glass of water to help neutralize stomach acid.

Note: Consult a healthcare professional before consuming baking soda for medicinal purposes, especially if you have underlying health conditions.

Eval Result:
1. Missing Steps: True
2. Redundant Steps: False
3. Duplicate Steps: False
4. Executable: True
5. Satisfy Constraints: False
6. Complete Goal: False
7. Step Order: True

Explain: The provided steps are missing the development process of an online calculator, which was the core task. Instead, they describe how to consume baking soda water, unrelated to creating a digital tool for calculating the recommended dosage based on individual body weight and age. There are no redundant or duplicate steps within the given script, but the entire script is off-target and does not address the development of an online calculator. The script does not meet the constraints because it does not include a mechanism to calculate the correct amount of baking soda based on individual body weight and age, which was a key requirement of the task. The step order is correct in terms of the process of preparing and consuming baking soda water, but the script as described does not include the initial calculation step necessary to achieve the goal.

Table 8: An example of eval result

Perturbed category: Missing Steps
Prompt: Please remove the most crucial steps from the script, so that the script cannot complete the goal. The goal of this script is [Goal]. The script is [Script]. Please answer my script after deleting the key steps.

Perturbed category: Redundant Steps
Prompt: Please add a redundant step that is completely unrelated to the target to the script. The goal of this script is [Goal]. The script is [Script]. Please provide a complete answer to the script I have added.

Perturbed category: Complete Goal
Prompt: Modify this script to make it CAN NOT achieve the goal [Goal]. The script is [Script]. Please provide me with the modified script.

Perturbed category: Satisfy Constraint
Prompt: Please modify this script to not meet the restrictions [Constraint]. The script is [Script]. Please answer me in this format. 2. 3. ...

Table 9: Prompt for adding perturbation to questions