TreeEval: Benchmark-Free Evaluation of Large Language Models through Tree Planning

Anonymous ACL submission

Abstract

Recently, numerous new benchmarks have been established to evaluate the performance of large language models (LLMs) via either computing a holistic score or employing another LLM as a judge. However, these approaches suffer from data leakage due to the open access of the benchmark and inflexible evaluation process. To address this issue, we introduce **TreeEval**, a benchmark-free evaluation method for LLMs that let a high-performance LLM host an irreproducible evaluation session and essentially avoids the data leakage. Moreover, this LLM performs as an examiner to raise up a series of questions under a topic with a tree planing strategy, which considers the current evaluation status to decide the next question generation and ensures the completeness and efficiency of the evaluation process. We evaluate 6 models of different parameter sizes, including 7B, 13B, and 33B, and ultimately achieved the highest correlation coefficient with AlpacaEval2.0 using only around 45 questions. We also conduct more analysis to show the robustness and reliability of TreeEval. Our code can be accessed via the provided URL[1].

1 Introduction

The recent surge in Large Language Models (LLM) has been significant, transitioning from closed-source (OpenAI, 2023; Team, 2023a) to open-source (Touvron et al., 2023; et al., 2023a; Jiang et al., 2023) models. Different Supervised Fine-Tuning (SFT) and Reinforcement Learning from Human Feedback (RLHF) techniques are further proposed to improve the performance of LLMs (Taori et al., 2023; Chiang et al., 2023; Bai et al., 2022; Ouyang et al., 2022; Tunstall et al., 2023a). These LLMs pose the capabilities to solve diverse tasks and are widely used in both academic and industrial fields. Human evaluation is intuitive to assess the performance of LLMs but it is time-consuming and has the risk of including unexpected bias (Zheng et al., 2023b; Wang et al., 2024). It becomes vital to investigate the automatic evaluation approaches to evaluate LLMs.

Till now, there are a number of automatic evaluation methods that have been proposed. A line of studies take efforts to annotate some benchmark datasets, such as MMLU, BBH, SocKET, AGIEval, IFEval and HalluQA (Hendrycks et al., 2021; Suzgun et al., 2022; Choi et al., 2023; Zhong et al., 2023; Zhou et al., 2023a; Cheng et al., 2023), to test the different capabilities of an LLM. The evaluated LLM is judged via checking the overlap between the annotated answers and generated answers. In this way, a holistic score is produced to indicate the performance of the LLM. We denote this category of evaluation methods are in a **benchmark paradigm**. However, the holistic score is inflexible to measure the quality of the outputs from LLMs because the token mismatch unnecessarily indicates the incorrect answer. With the emergence of high-performance LLMs, another line of studies start to leverage them to simulate human evaluation by providing the evaluated LLM with pre-defined benchmark questions and requesting another LLM like GPT-4 to judge its response (Zheng et al., 2023a; Li et al., 2023b; Bai et al., 2023; Wang et al., 2023a; Zhang et al., 2023b; Wang et al., 2023c; Li et al., 2023a; Zhu et al., 2023). We denote this category of evaluation methods as **LLM-as-judge paradigm**. Even though both above lines of methods enable the automatic evaluation with standard pipelines, they encounter severe issue of data leakage. Due to the vast amount of training data used in LLM training, considered a valuable asset by many closed and even open-source models, it is easy to cause benchmark data leakage problems and significantly biases evaluation results.

To solve this issue, we propose a novel evaluation paradigm, which takes an LLM as an examiner to raise questions. The examiner should

[1] Anonymous github repository

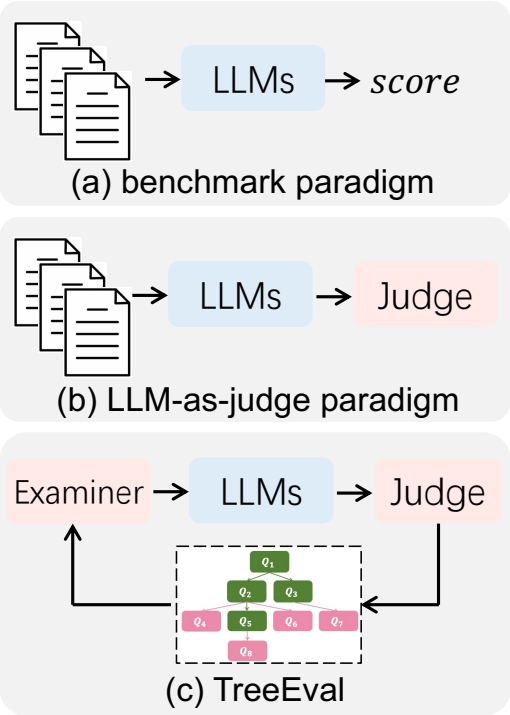

Figure 1: Comparison of TreeEval with existing evaluation paradigms.

produce different evaluation session for each time which makes it hard to duplicate the evaluation questions and protect the evaluation benchmark from disclosure for fine-tuning and pre-training an LLM deliberately. However, simply adopting an LLM as examiner would lead to arbitrary evaluation question generation without a goal. Designing such a benchmark-free evaluation method need take the following aspects into consideration: (1) Similar as the question in a benchmark (Taori et al., 2023; Zheng et al., 2023a), the generated questions should be derived from certain topics, which ensures the scope of the evaluation. (2) Drawing inspiration from the interview, within a topic, the examiner should generate a line of questions that are diverse to cover different knowledge rather than producing a single question. (3) The generation procedure should be flexible enough to generate mutually connected questions and control the difficulty level of these questions. When the current line of question cannot distinguish two LLMs, more difficult questions should be raised up. Otherwise, the evaluation could be terminated immediately.

To this end, we propose **TreeEval**, which is a benchmark-free evaluation of LLMs through tree planning. The line of questions within a topic for evaluation are organized in a tree, where each node contains a question. In the process of constructing a tree, we repeatedly revisit the status of the current tree and generate the next node until the tree is enough to differentiate two LLMs. The difference between our evaluation method and previous paradigms can be found in Figure 1. To verify the effect of our method, we evaluate multiple LLMs. The results demonstrate that our method shows similar ranking as AlpacaEval2.0 in LLM-as-judge paradigm with only 45 questions in average for each round of evaluation. Further analysis shows our advantages in measuring fine-grained capabilities and conducting robust comparison for LLMs.

Our contributions are summarized as follows:

- We introduce a novel evaluation paradigm, TreeEval, which allows for efficient and comprehensive evaluation of LLMs, inherently preventing data leakage issues.
- TreeEval has advantage in distinguishing two LLMs with similar performance by constructing a deeper tree, which extends the evaluation process to obtain more stable and accurate assessment results.
- We compare with a set of automatic evaluation baselins, and find that our TreeEval achieves the highest correlation coefficient with AlpacaEval2.0.

2 Related Work

2.1 Methods of LLM Evaluation

Due to the explosive growth and rapid update of LLMs, a significant challenge is to conduct accurate and comprehensive evaluation for them (Chang et al., 2023). Early studies leverage open-ended question answering datasets and math word problems as the evaluation benchmarks (Touvron et al., 2023; et al., 2023b; Chen et al., 2024) to evaluate the commonsense knowledge and reasoning capabilities of LLMs. Subsequently, more benchmark datasets like MMLU (Hendrycks et al., 2021), AGIEval (Zhong et al., 2023), IFEval (Zhou et al., 2023a) have been elaborately designed to gauge diverse abilities of LLMs. Some studies (Wang et al., 2023a,a; Saha et al., 2023) go beyond standard evaluation metrics. They evaluate the quality and accuracy of predicted results through human annotation, which is able to provide a more comprehensive feedback. With the emergence of high-performance LLMs like GPT-4 (OpenAI, 2023),

Gemini Pro (Team, 2023a), more recent studies start to utilize them to simulate the human evaluation process. In this realm, PandaLM (Wang et al., 2023c) strives to provide reproducible and automated comparison between various LLMs by training a LLM as the judge. GPTScore (Fu et al., 2023) and G-Eval (Liu et al., 2023) utilize GPT-3 and GPT-4 as the judge to evaluate the LLMs with incorporation of in-context learning and chain-of-thought strategies. The above methods rely heavily on a well-organized benchmark dataset while our method is benchmark-free and has LLMs performing as the examiner.

2.2 Data Leakage of LLM Evaluation

As the number of benchmarks for language model evaluation increases, data leakage emerges as an inevitable concern. However, there appear to be a limited number of studies addressing this issue. Sainz et al.(2023) propose a method to detect data breaches in closed-source LLMs, based on the premise that LLMs can recall training data and tend to reproduce similar content. Zhou et al.(2023b) conduct qualitative analysis of the impact of data leakage, which suggests that a data breach in one benchmark significantly enhances the LLM's performance on that specific benchmark while diminishing its capabilities on other uncompromised benchmarks. Yang et al.(2023) propose a more accurate approach which employs an LLM detector with top-k closest training data to determine if they match the test data. In contrast to these methods, which develop additional models for detecting data leakage during LLM evaluation with given benchmark datasets, our proposed method introduces a novel paradigm for LLM evaluation. It not only ensures the high quality of test questions but also inherently avoids data leakage.

3 Methodology

3.1 Overall Architecture

Figure 2 depicts the overall architecture of TreeEval. To organize the evaluation in a tree structure, TreeEval incorporates several components including an *Examiner*, a *Judge*, and an *Eval Controller*. After the tree has been constructed, an *Aggregator* is utilized to aggregate the scores in the tree. The entire framework achieves benchmark-free evaluation of LLMs with a process of tree planning.

In detail, for each session of evaluation, we choose a pair of LLMs for evaluation. The evaluation process starts with an initial topic. The examiner takes charge of generating questions within a given topic. Then, the question will be sent to the pair of LLMs under test to produce the responses. Next, the judge compares the responses from the two LLMs and decides the winner for this question. Considering the responses for the current question, the eval controller will determine to deepen the current question if the current question results in a close tie or terminate the search along the current question if it is clear to decide the winner. We employ a breadth-first search strategy for this process. When we generate the next question, the eval controller also guarantees the diversity and reliability of the questions. Eventually, the nodes traversing over the tree will be aggregated to produce a comprehensive score for the evaluation.

TreeEval can be viewed as a tree-planning framework using LLMs as heuristic for hosting the evaluation session. By constructing a tree structure, we can differentiate the abilities of a pair of LLMs using as few as questions. Moreover, the test questions are automatically generated during each evaluation, which prevents the benchmark leakage. In the tree, the root node is initialized by the identified topic for the session. Each node contains a topic and a question within the topic. The connection between nodes indicates inheritance relationships between the questions, that is, the deepen question is derived from its parent question. The deeper the tree is, the more similar abilities the two LLMs under test hold. For sibling nodes in the tree, which are derived from the same parent node, they are designed to cover distinct topics but belong to the same superior topic.

3.2 TreeEval Modules

In this section, we provide more details of the components of the TreeEval and illustrate how to construct a tree for evaluation via these components.

Examiner. The examiner is a LLM-based module, which takes charge of generating exam questions that are able to cover diverse topics. Following (Bai et al., 2023), we pre-define a set of topics as the scope of evaluation.

As the initialization of an evaluation session, we randomly sample a topic from the pre-defined topic set, which is denoted as $\mathcal{FC}_{\text{pre-define}}$. Given a topic, the examiner is requested to craft a question that related to it via a prompt with the consideration of the coherence to the topic and the required format of the question. The detailed instruction is displayed

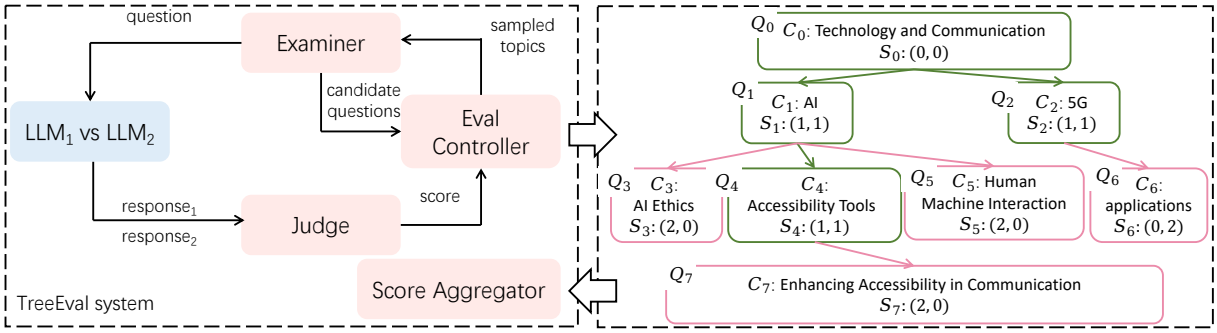

Figure 2: TreeEval system with an illustrative tree for evaluation. The left section contains the components and their workflow in TreeEval. The right section displays a constructed tree within topic *Technology and Communication* for evaluation (the leaf nodes are shown in red boxes), where each node denotes a question annotated with its topic and evaluation score. We further display the generated questions of the tree in the Appendix 8.6.

in Appendix 8.1.

Once the session begins, we organize the follow-up questions in a tree structure. For simplify, we generally denote the follow-up topic at the t-th time step as C_t. And the above procedure can be presented as:

$$Q_t = \text{Examiner}(C_t).$$

Subsequently, Q_t is utilized as the question to test the LLMs under review.

Judge. Previous studies (Wang et al., 2023a) conduct pare-wise comparison and identify the superior responses among two evaluated LLMs, which has advantage in providing more nuanced assessment. Following these studies, we consult a pair of LLMs with the same question. The detailed instruction is displayed in Appendix 8.2. After the responses have been produced via the LLMs, another LLM performs as the judge to the responses.

To ensure the reliability of the judge, we further conduct exchange evaluation, that is to switch the order of the responses. This procedure can be denoted as:

$$S_t^1 = \text{Judge}(Q_t, A_t^1, A_t^2);$$
$$S_t^2 = \text{Judge}(Q_t, A_t^2, A_t^1),$$

where A_t^1, A_t^2 denote the responses from the pair of LLMs for Q_t. Each output judges the winner is A_t^1 or A_t^2 or a tie exits. If there is an agreement for S_t^1 and S_t^2, We assign 2 score to the winner and 0 score to the loser to form S_t. Otherwise, we assign 1 to each model as S_t.

As the evaluation proceeds, we maintain a memory to record the history of the session, including the initial topic, historical questions as well as responses from the two evaluated LLMs. After Q_t has been responded, the history at the t-th time step in the evaluation session can be denoted as $\mathcal{M}_t = \{C_0, Q_0, A_0^1, A_0^2, ..., C_t, Q_t, A_t^1, A_t^2\}$. To involve the coherence of the flowing conversation and raise up rational follow-up questions, we prompt the examiner with the consideration of the history.

Eval Controller. The evaluation controller takes charge of the process of tree planning. Arbitrary generation of questions result in unorganized evaluation of LLMs with repeated questions and limited topics. To ensure the relevance and diversity of the generated questions, we have the following consideration: (1) To simulate the real-world interview of a certain subject, where the questions in an examination are mutually connected, we assume the generated follow-up question should be closely linked to its previous question via topics. For example, in Figure 2, inheriting from the root topic "*technology and communication*", we can raise a question on "*5G*" that is relevant to the root topic and goes deeper. (2) The generated questions should not be repeated in the existing questions and we should ensure the diverse knowledge covered by the tree. For example, in Figure 2, under the topic "*AI*", we can come up with distinct but related sub-topics as siblings such as "*AI Ethics*", "*Accessibility Tools*" and "*Human Machine Interaction*".

Inspired by the Tree-of-Thought (Long, 2023), where a controller produces the next thought step, we let Eval Controller arrange the follow-up evaluation according to $\mathcal{M}_t$. On the one hand, it prepares the follow-up topics $\mathcal{FC}_t$ based on $\{C_t, A_t^1, A_t^2\} \in \mathcal{M}_t$ for any of its child nodes in advance. On the other hand, it determines Q_{t+1} based on the $\mathcal{FC}_t$ and $\{Q_1, Q_2, ..., Q_t\} \in \mathcal{M}_t$ if the t-th node is the parent node at $t+1$ time step. We next describe

Algorithm 1 Procedure of TreeEval

Input $\mathcal{FC}_{\text{pre-define}}$;
Initial $t \leftarrow 0$; $\mathcal{M}_t \leftarrow \emptyset$
while Termination strategy is not satisfied **do**
  for $C_t \in \mathcal{FC}_{parent}$ or $C_0 \in \mathcal{FC}_{\text{pre-define}}$ **do**
    $\tilde{\mathcal{Q}}_t \leftarrow \text{Examiner}(C_t)$ ▷ Sample questions via Examiner.
    $Q_t \leftarrow \arg\max_{Q_t^i \in \tilde{\mathcal{Q}}_t}(\text{Sim}(Q_t^i, C_t) - \max_{Q_k \in \mathcal{M}_t} \text{Sim}(Q_t^i, Q_k))$ ▷ Rank candidate questions in Step Two.
    $A_t^1, A_t^2 \leftarrow \text{LLMs}(Q_t)$
    $S_t = \text{Judge}(Q_t, A_t^1, A_t^2)$ ▷ Output scores of LLMs via Judge.
    $\mathcal{M}_t \leftarrow \mathcal{M}_t \cup \{C_t, Q_t, A_t^1, A_t^2\}$
    $\tilde{\mathcal{FC}}_t \leftarrow \text{NER}(A_t^1) \cup \text{NER}(A_t^2)$ ▷ Generate candidate topic in Step Two.
    $\mathcal{FC}_t \leftarrow \emptyset$
    while $|\mathcal{FC}_t| < k$ **do** ▷ Iteratively Filter candidate topics in Step One.
      $C_t^i \leftarrow \arg\max_{\tilde{C}_t^i \in \tilde{\mathcal{FC}}_t}(\text{Sim}(\tilde{C}_t^i, C_t))$
      $\mathcal{FC}_t \leftarrow \mathcal{FC}_t \cup \{C_t^i\}$
      $\tilde{\mathcal{FC}}_t \leftarrow \tilde{\mathcal{FC}}_t \setminus C_t^i$
      for $\tilde{C}_t^j \in \tilde{\mathcal{FC}}_t$ **do**
        $\text{Sim}(\tilde{C}_t^j, C_t) \leftarrow \text{Sim}(\tilde{C}_t^j, C_t) - \text{Sim}(\tilde{C}_t^j, C_t^i)$
    $t \leftarrow t + 1$

the above two steps in detail:

- **Step One**: Sample topics from the responses of the previous question: $\mathcal{FC}_t \sim \text{NER}(A_t)$[2]. This works better when the Named Entity Recognition (NER) tool is built upon a LLM as some relevant entities could be revised via the model instead of solely being extracted (Wang et al., 2023b).

  We sample candidate topics from both A_t^1 and A_t^2 then merge them together, which results in a set of candidate topics $\tilde{\mathcal{FC}}_t$ as the follow-up topics of t-th node. However, this may produce some candidates that are repeated. To avoid this, we first measure the similarity between $C_t^i \in \tilde{\mathcal{FC}}_t$ and C_t by computing the Cosine Similarity of their encoded vector representation (Zhang et al., 2023a), which is denoted as $\text{Sim}(C_t^i, C_t)$. Then, we iteratively push out C_t^i with the largest score. Next, we update the similarity scores of the rest topic C_t^j by subtracting the similarity score of $C_t^j \in \tilde{\mathcal{FC}}_t \setminus C_t^i$ and C_t^i, which is to decrease the possibility of retrieving similar topics. This procedure continues until we have pushed out k topics as $\mathcal{FC}_t$ for the follow-up question generation.

- **Step Two**: If the question at $(t+1)$-th time step is the child node of the node at t-th time step, we generate questions based on the sampled topic via $Q_{t+1}^i \sim \text{Examiner}(C_{t+1})$, where $C_{t+1} \in \mathcal{FC}_t$. This could form a candidate question set $\tilde{\mathcal{Q}}_{t+1}$. Still, to avoid repetition of the generated questions and ensure a broad spectrum of inquiry questions, we conduct ranking for the candidate questions. Specifically, we measure the similarity between $Q_{t+1}^i \in \tilde{\mathcal{Q}}_{t+1}$ and C_{t+1} via Cosine Similarity. Then we push out Q_{t+1}^i with the largest similarity score of $\text{Sim}(Q_{t+1}^i, C_{t+1})$ and the least similarity score of $\arg\min_{Q_k \in \mathcal{M}_t} \text{Sim}(Q_{t+1}^i, Q_k)$.

Termination Strategy. To determine whether we should stop generating subsequent questions along a topic, we identify several termination criteria:

- For each node in the tree, if the question posed by the current topic successfully distinguish the capability of the two LLMs under review. Alternatively, if there is no tie for the current question, we terminate the child node search under the current node.

- After we have generated the sibling nodes of a parent node, we revisit the the scores of these siblings. If it shows a dominate score over all these siblings, this indicates that we have a winner for this branch. Hence we stop further search for any of these sibling nodes.

[2]The detailed instruction could be found in Appendix 8.3

- To prevent the evaluation session preceding indefinitely, a maximum depth T for the tree search is pre-defined. Once the limit reaches, we terminate the child node search under the current topic.

We terminate a tree search when every node in the tree satisfies the above criteria. The entire process is described in Algorithm 1.

3.3 Score Aggregator

After we have constructed the multiple trees across $\mathcal{FC}_{\text{pre-define}}$, where the nodes in each tree implies the win-rate between two LLMs under review towards a specific topic. To yield a final win-rate result, we aggregate the scores of these constructed trees. However, it is irrational to consider all the nodes in a tree equally due to their different features and result scores. Specifically, we take the following aspects of t-th node in a tree into account when we aggregate their scores:

- **Distance to the root node.** Based on the principle of an evaluation session, a longer distance to the root node indicates a more intensive competition between the evaluated LLMs and the more important the node is. This suggests that the winner only has a marginal advantage over the other one. Therefore, we define one aspect of an important node as $w_t^{\text{root}} = \frac{1}{d}$, where d is the distance from the t-th node to the root node in a tree.

- **Origin of the topic.** As the topic is derived from the responses in its parent node, a node inherited the topic generated from responses of the losing LLM is more important considering it is more likely to balance the situation. Hence, we define one aspect of an important node as:

$$w_t^{\text{topic}} = \begin{cases} 1 & \text{Topic originated from the loser} \\ 0.5 & \text{Otherwise} \end{cases}$$

- **Variance of the sibling nodes.** The disagreement of the evaluation of the sibling node may implicit a potential randomness derived from the topic. So we define the sibling consensus as:

$$w_t^{\text{topic}} = \frac{1}{\sigma^2 + 1},$$

where σ is the variance of the score of its sibling nodes.

Considering the above aspects, we compute the final importance weights of t-th node as:

$$w_t = {w_t^{\text{root}}}^{\alpha} \cdot {w_t^{\text{topic}}}^{\beta} \cdot {w_t^{\text{sibling}}}^{\gamma},$$

where α, β, and γ are hyper-parameters indicating the relative importance of these aspects. As a result, we sum up the w_t multiplying with the win-rate of an LLM and devide the total evaluation questions to obtain its final scores:

$$S = \frac{1}{N} \sum_{i\text{-th Tree from } \mathcal{FC}_{\text{pre-define}};\ t\text{-th node in } i\text{-th Tree}} w_t \cdot S_t,$$

where N is the sum of node weights in the evaluation session and S is normalized.

4 Experiments

4.1 Experimental Setup

Evaluated LLMs. We consider the following open-source LLMs as evaluated LLMs, encompassing two 7B models, two 13B models, and two 33B models. They are either derived from LLaMA (Touvron et al., 2023; et al., 2023a) or trained from scratch using the LLaMA architecture and some of them exhibit remarkably similar performances according to the open-source LLM leader-board[3].

In particular, `Yi-34B-Chat` (01.AI, 2023) is a pioneering product from 01.AI. It is built on a large-scale multilingual dataset. `Xwin-LM-13B-V0.1` (Team, 2023b) is built on the foundation of LLaMA2-13B, tuned through SFT and RLHF. `Mistral-7B-Instruct-v0.2` (Jiang et al., 2023) is tuned through SFT on the Mistral-7B model, which is built with the LLaMA architecture. `Vicuna-33b-v1.3` (Zheng et al., 2023a) originates from the LLaMA-33b model and is fine-tuned using user-shared dialogues from ShareGPT. `WizardLM-13B-V1.2` (Xu et al., 2023), based on the LLaMA2-13b model, is fine-tuned with enhanced instruction data using the Evol-Instruct. `Zephyr-7B-beta` (Tunstall et al., 2023b), derived from the Mistral-7B model, is algined with SFT and DPO methods.

Comparable Evaluation Methods. We compare TreeEval with a series of existing evaluation methods. We include multiple methods with benchmark paradigm: **MMLU** (Hendrycks et al., 2021), and Big-Bench Hard (Suzgun et al., 2022) (**BBH**). Meanwhile, we include multiple methods treating LLMs as judges: **AlpacaEval**, **AlpacaEval2.0** (Li et al., 2023b), and **MT-Bench** (Zheng et al., 2023a), which all choose ChatGPT as the judge but differ in their handling of dialogue scenarios. Specifically, MT-Bench is designed to assess multi-turn

[3] https://tatsu-lab.github.io/alpaca_eval/

LLMs	MMLU*	BBH*	AlpacaEval†	MT-bench†	AlpacaEval2.0†	TreeEval(Ours)	
	Acc	Acc	Win-Rate	score	Win-Rate	#Q	Score
`Mistral-7B-Instruct-v0.2`	70.6	46.4	92.78	8.30	14.72	–	2.50
`Yi-34B-Chat`	73.46	71.74	94.08	8.65	29.66	31.67	3.48
xwinlm-13b-v0.1	56.6	37.58	91.76	7.34	17.43	62.33	2.67
`Vicuna-33b-v1.3`	59.2	52.0	88.99	7.12	12.71	41.33	1.61
`WizardLM-13B-V1.2`	52.7	40.12	89.17	7.2	12.03	44.67	1.10
`zephyr-7b-beta`	61.4	42.72	90.60	7.34	10.99	45.67	2.19
Average #Q ↓	14,079	6,511	804	80	804	**45.1**	–
ρ ↑	0.43	0.37	0.71	0.61	1.0	–	**0.83**
τ ↑	0.33	0.33	0.47	0.41	1.0	–	**0.73**

Table 1: Comparison of LLMs across various evaluation methods. "$\star$" denotes we re-implement MMLU and BBH benchmarks (Chia et al., 2023), calculating results in both 5-shot and 3-shot contexts. "†" denotes we directly take results from the respective leader-boards from MT-bench, AlpacaEval, and AlpacaEval2.0. "#Q" denotes the number of questions used for evaluation. We report the correlation of rankings obtained through different methods with those from AlpacaEval2.0, using τ for the Kendall correlation coefficient (KENDALL, 1938) and ρ for the Spearman correlation coefficient (Spearman, 1904).

dialogues, whereas AlpacaEval focuses solely on evaluating single-turn interactions.

Implementation Details. We utilize `GPT-4-0613` as our examiner and deploy the model using FastChat (Zheng et al., 2023a). We set the temperature parameter to 1, which facilitates to generate variant questions. Furthermore, we set T and k as 3. α, β, and γ are set as 1, 1, 0.4, respectively. To mitigate the randomness of the experiments, we repeat 3 times and calculate the average scores. We choose `Mistral-7B-Instruct-v0.2` as the reference for pairwise comparison due to its moderate performance shown in public leaderboard.

4.2 Performance of TreeEval

We display the performance of TreeEval in Table 1, from which we have the following observations: (1) Among all the comparable evaluation methods, our method is able to achieve the highest correlation coefficient with the rankings of AlpacaEval2.0 on the indicators of both ρ and τ. AlpacaEval2.0 is commonly viewed as the recognized LLM evaluation leader-board and the high consistency between our ranks indicates the reliability of our method. (2) Our method is able to complete the evaluation procedure with only 45 questions in average while the other evaluation methods require much more questions to generate an evaluation result. This indicates that our evaluation is efficient on evaluating LLMs with minimum questions. (3) Since we treat `Mistral-7B-Instruct-v0.2` as the reference for the pairwise comparison, we notice the larger gap between the evaluated LLM and the reference is, the less test questions are proposed in the evaluation session, which shows that tree planning indeed meets our expected motivation. We further display the pairwise correlation in Appendix 8.5 to show the correlation between TreeEval and AlpacaEval is also high.

4.3 Further Analysis

We conduct further analysis to verify the effect of TreeEval[4].

Fine-grained Evaluation. From the Figure 4, we can observe that different LLMs excel in various knowledge domains. The performance of the same model may vary across different fields. For example, `Yi-34B-Chat` is short in *Travel and Shopping* while it demonstrates relatively good performance on other topics. Through our TreeEval, we can diagnose an LLM with the fine-grained results in diverse domains.

Robustness of TreeEval. We draw Figure 5 to show the evaluation results of different LLMs in multiple runs. We can see that, for a given LLM under evaluation, conducting multiple repeated experiments yields relatively similar scores when the examiner's temperature is set to 1. The low variance indicates that TreeEval is able to generate stable and robust results.

[4]More pairwise comparison for different model pairs and method pairs analysis can be found in Appendix 8.4,8.5.

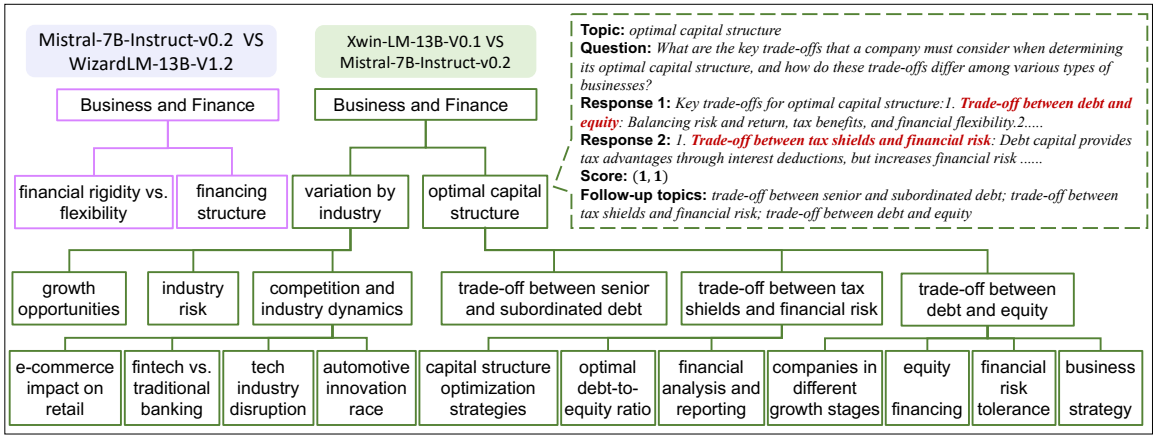

Figure 3: Examples of evaluation process for two pairs of LLMs under topic "*Business and Finance*", which are shown in two colored trees. The detailed contents of a node is displayed in a dashed box and the recognized entities used for follow-up topics are shown in red fonts.

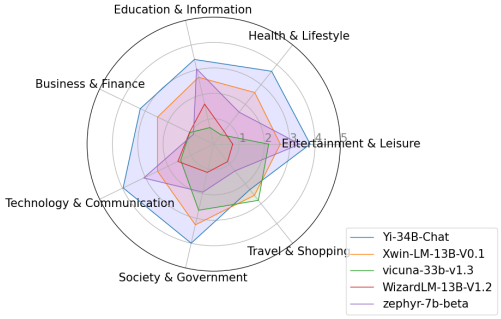

Figure 4: Radar chart illustrating the scores of various LLMs under different pre-defined topics.

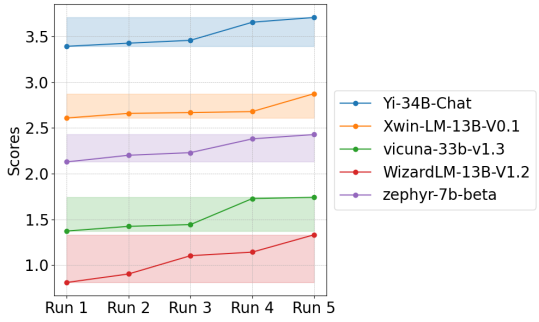

Figure 5: Re-run TreeEval 5 times for various LLMs.

Ablation Studies. As we can see in Table 2, changing BFS search to DFS search dramatically increases the number of questions but decreases the performance. This is because DFS search generates the child node first rather than the sibling node such that the influence of sibling node will be neglected in both question generation and termination identification procedures. Removing step one, which indicates skip the topic generation step, decreases the performance. This indicates the significant role of identifying the topic for question generation. When we iteratively remove the scores in aggregator, we observe general performance drop on τ. This indicates that all the scores in the aggregator are important in producing a comprehensive score.

Methods	#Q	ρ	τ
TreeEval	45.1	0.83	0.73
BFS $\rightarrow$ DFS	149.4	0.37	0.33
w/o Step One	49.3	0.31	0.2
w/o w^{root}	45.1	0.77	0.6
w/o w^{topic}	45.1	0.77	0.6
w/o w^{sibling}	45.1	0.71	0.47

Table 2: Ablation study on TreeEval.

Case Studies. In Figure 3, it's clear that TreeEval effectively identifies performance gaps between LLMs. For instance, `Mistral-7B-Instruct-v0.2` notably outperforms `WizardLM-13B-V1.2`, reflected in a smaller tree. Conversely, when models perform similarly, like `Mistral-7B-Instruct-v0.2` and `Xwin-LM-13B-V0.1`, TreeEval constructs a larger tree to discern subtle performance differences.

5 Conclusions

In this paper, we introduce TreeEval, a benchmark-free evaluation approach for LLMs with tree planning, which automatically controls the evaluation process with tree planning. We experimentally verify that TreeEval can not only produce reliable evaluation results without data leakage but also enhance discrimination between similarly performing LLMs.

6 Ethical Considerations

Although we prioritize the security of the LLMs we use during evaluations, striving to employ aligned LLMs with higher safety standards, and endeavor to ensure that LLM outputs adhere to ethical and legal requirements, limitations arising from model size and probabilistic generation paradigms may lead to various unexpected outputs. These could include questions or responses containing biases, discrimination, or other harmful content. Please refrain from disseminating such content.

7 Limitations

While we utilize the most powerful general LLM, GPT4, as our examiner in the evaluation process, it is important to recognize that even GPT4 has its limitations and areas where it may not excel. One inherent flaw in our approach is that it may fail when evaluating content that the examiner isn't proficient in. One potential solution could involve providing more contextual cues to guide the examiner when assessing challenging content. Looking ahead, training a specialized examiner to extract questions from a document repository and evaluate an LLM's comprehension of the knowledge within that repository could address the issue of evaluating content that the examiner isn't proficient in.

References

01.AI. 2023. Yi.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yushi Bai, Jiahao Ying, Yixin Cao, Xin Lv, Yuze He, Xiaozhi Wang, Jifan Yu, Kaisheng Zeng, Yijia Xiao, Haozhe Lyu, Jiayin Zhang, Juanzi Li, and Lei Hou. 2023. Benchmarking foundation models with language-model-as-an-examiner. *arXiv preprint arXiv:2306.04181*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2023. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*.

Hailin Chen, Fangkai Jiao, Xingxuan Li, Chengwei Qin, Mathieu Ravaut, Ruochen Zhao, Caiming Xiong, and Shafiq Joty. 2024. Chatgpt's one-year anniversary: Are open-source large language models catching up? *arXiv preprint arXiv:2311.16989*.

Qinyuan Cheng, Tianxiang Sun, Wenwei Zhang, Siyin Wang, Xiangyang Liu, Mozhi Zhang, Junliang He, Mianqiu Huang, Zhangyue Yin, Kai Chen, and Xipeng Qiu. 2023. Evaluating hallucinations in chinese large language models. *arXiv preprint arXiv:2310.03368*.

Yew Ken Chia, Pengfei Hong, Lidong Bing, and Soujanya Poria. 2023. Instructeval: Towards holistic evaluation of instruction-tuned large language models. *arXiv preprint arXiv:2306.04757*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Minje Choi, Jiaxin Pei, Sagar Kumar, Chang Shu, and David Jurgens. 2023. Do llms understand social knowledge? evaluating the sociability of large language models with socket benchmark. *arXiv preprint arXiv:2305.14938*.

Hugo Touvron et al. 2023a. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Rohan Anil et al. 2023b. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

M. G. KENDALL. 1938. A NEW MEASURE OF RANK CORRELATION. *Biometrika*, 30(1-2):81–93.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2023a. Generative judge for evaluating alignment. *arXiv preprint arXiv:2310.05470*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

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

Jieyi Long. 2023. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. 2023. Branch-solve-merge improves large language model evaluation and generation. *arXiv preprint arXiv:2310.15123*.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787, Singapore. Association for Computational Linguistics.

C. Spearman. 1904. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. *GitHub repository*.

Gemini Team. 2023a. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Xwin-LM Team. 2023b. Xwin-lm.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. 2023a. The alignment handbook. `https://github.com/huggingface/alignment-handbook`.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023b. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Binghai Wang, Rui Zheng, Lu Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, Songyang Gao, Nuo Xu, Yuhao Zhou, Xiaoran Fan, Zhiheng Xi, Jun Zhao, Xiao Wang, Tao Ji, Hang Yan, Lixing Shen, Zhan Chen, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. 2024. Secrets of rlhf in large language models part ii: Reward modeling. *arXiv preprint arXiv:2401.06080*.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023a. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023b. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2023c. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. 2023. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*.

Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. 2023a. Retrieve anything to augment large language models. *arXiv preprint arXiv:2310.07554*.

Xinghua Zhang, Bowen Yu, Haiyang Yu, Yangyu Lv, Tingwen Liu, Fei Huang, Hongbo Xu, and Yongbin Li. 2023b. Wider and deeper llm networks are fairer llm evaluators. *arXiv preprint arXiv:2308.01862*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023a. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Cheng Chang, Zhangyue Yin, Rongxiang Weng, Wensen Cheng, Haoran Huang, Tianxiang Sun, Hang Yan, Tao Gui, Qi Zhang, Xipeng Qiu, and Xuanjing Huang. 2023b. Secrets of rlhf in large language models part i: Ppo. *arXiv preprint arXiv:2307.04964*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *arXiv preprint arXiv:2304.06364*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023a. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023b. Don't make your llm an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964*.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2023. Judgelm: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*.

8 Appendix

8.1 Prompt for Examiner

I want you to assume the role of the expert and ask a question that expands and reflects your understanding of {topic}. Your task is to ask a question about {topic}. Only through a profound understanding of {topic} can one correctly answer this question. Please adhere strictly to the following 4 task guidelines:

1. Your question should begin with a question word, such as "what", "which", "when", "where", "how", "why", etc.

2. The objective of your question should be to manifest the respondent's understanding of {topic} and to differentiate respondents based on their comprehension level.

3. Questions should be self-explanatory, not requiring additional context or clarification.

4. Please format your question in the JSON structure provided below. Remember, only output the content in the following format, and nothing else: {{"question": your question}}

8.2 Prompt for Judge

You are assessing two submitted responses to a user's query based on specific criteria. Evaluate the quality, relevance, accuracy, clarity, and any other relevant factors to determine which response is superior, or if they are equally valuable or lacking. Here is the data for your assessment:
[Query]: {question}
[Response 1]: {answer 1}
[Response 2]: {answer 2}
Assessment Criteria:

1. Relevance to the query: Does the response directly address the user's question or concern?

2. Accuracy of information: Are the facts or solutions provided in the response correct and reliable?

3. Clarity and comprehensibility: Is the response easy to understand, well-structured, and free of jargon or ambiguity?

4. Completeness: Does the response cover all aspects of the query or offer a comprehensive solution?

5. Additional value: Does the response provide extra insights, tips, or information that enhances the user's understanding or solves the problem more effectively?

Instructions for Assessment:

1. Identify and focus on the criteria that significantly distinguish the two responses. Disregard criteria that do not offer a clear distinction.

2. Consider any specific aspects of the query and the responses that may require additional factors for a fair comparison. Mention these factors explicitly.

3. Conclude your assessment by deciding which response is better, or if they are tied. Your decision must be based on a coherent evaluation across the mentioned criteria and any additional factors you've identified.

Please return your final decision in the following JSON format: {"Eval_result": "Response 1"/"Response 2"/"Tie"}
Note: Remember, the output should only contain the decision in the specified JSON format and nothing else.

8.3 Prompt for NER

You are asking questions and answers based on a topic you know and based on this topic. Please extract some subtopics from the answers. Here's an example:

Here is the data:
[Input data]
***
[topic]: programming languages
***
[question]: Which programming languages can you write code in?
***
[answer]: I know python, C++, R language, etc.
***
[Output Data]
***
[subtopic] : ["python","C++","R language"]
now the official question
Here is the data:
[Input data]
***
[topic]:0
***
[question]:1
***
[answer]:2
***
[Output Data]
***

Please return your final decision in list format. Remember, you only need to output the content in the following List format, with each element as a subtopic and nothing else. Remember, you only need to output the three most important subtopics in the following List format.

[subtopic] : ["subtopic1","subtopic2","subtopic3"]

8.4 Pairwise Comparison for different model pairs

Model	Yi-34B -Chat	Xwin-LM -13B-V0.1	Mistral-7B -Instruct-v0.2	vicuna -33b-v1.3	WizardLM -13B-V1.2	zephyr -7b-beta
Yi-34B-Chat	–	1.88	1.52	2.1	1.21	1.75
Xwin-LM-13B-V0.1	3.12	–	2.33	1.53	1.57	2.41
Mistral-7B-Instruct-v0.2	3.48	2.67	–	1.61	1.10	2.19
vicuna-33b-v1.3	2.9	3.47	3.39	–	2.01	3.7
WizardLM-13B-V1.2	3.79	3.43	3.90	2.99	–	3.94
zephyr-7b-beta	3.25	2.59	2.81	1.3	1.06	–

Table 3: Our result for each model pairs. The elements in this table represent the scores obtained by comparing models using treeEval, with the column model being compared against the row model. A score greater than 2.5 indicates that the model corresponding to the column outperforms the model corresponding to the row.

We iteratively change the references for the pairwise comparison and the results are shown in Table 3. Choosing the right baseline model is a critical step in our evaluation strategy. We selected the `Mistral-7B-Instruct-v0.2` as our baseline, emphasizing the significance of selecting a baseline that accurately reflects the broad insights from pairwise model comparisons. Ideally, a baseline model should

have a performance level that is neither too high nor too low, ensuring fair and balanced comparisons across all models. Interestingly, our observations indicate that even a baseline model chosen at random can lead to rankings that closely resemble those from a thorough pairwise evaluation. Thus, it's feasible to start with a randomly chosen baseline model to set up an initial order of performance. This preliminary order can be refined effectively using the insertion sort method. Given the initial order's similarity to the final ranking, this refinement process tends towards an $O(n)$ complexity, significantly enhancing the evaluation's precision and efficiency.

8.5 Pairwise Correlation for different method pairs

	MMLU		BBH		AlpacaEval		MT-bench		AlpacaEval2.0		TreeEval (Ours)	
	ρ	τ	ρ	τ	ρ	τ	ρ	τ	ρ	τ	ρ	τ
MMLU	–	–	−0.03	−0.07	0.14	0.07	0.09	0	0.43	0.33	0.49	0.33
BBH	−0.03	−0.07	–	–	0.77	0.6	0.84	0.69	0.37	0.33	0.49	0.33
AlpacaEval	0.14	0.07	0.77	0.6	–	–	0.99	0.97	0.71	0.47	0.89	0.73
MT-bench	0.09	0	0.84	0.69	0.99	0.97	–	–	0.61	0.41	0.81	0.69
AlpacaEval2.0	0.43	0.33	0.37	0.33	0.71	0.47	0.61	0.41	–	–	0.83	0.73
Ours	0.49	0.33	0.49	0.33	0.89	0.73	0.81	0.69	0.83	0.73	–	–

Table 4: Correlation coefficients of model rankings calculated using different lists.

As the tabel 4 shown that the ranks given by benchmark-based paradigm (i.e., MMLU and BBH) are similar and the ranks derived from LLMs-as-judge paradigm are similar (AlpacaEval, MT-bench, AlpacaEval2.0, and TreeEval). Our TreeEval has the ranking that is relatively close to AlpacaEval and AlpacaEval2.0.

8.6 Eval Controller Example

For the first aspect: the generated follow-up question should be closely linked to its previous question via topics. For example, in Figure 2, we generate Q_2 as "How does 5G technology improve Internet of Things (IoT) applications and smart city initiatives?" Based on the C_6 we obtained we generated a new question Q_6 "In what ways does 5G technology enable advancements in smart home devices and automation?", Q_6 is an expansion and extension of Q_2 on C_6 "applications".

For the second aspect: the generated questions should not be repeated in the existing questions. For example, in Figure 2, Q_5 was initially "What are the ethical considerations when designing AI systems for human-machine interaction?" However, this significantly overlapped with our previously established Q_3 "What are the key ethical considerations when developing AI technologies for communication platforms?" Consequently, we opted for Q_5 to be "What role does AI play in the development of voice-activated systems, and how does it change human-machine interaction?" to ensure variety and specificity in our discussion topics.