Landscape of Thoughts: Visualizing the Reasoning Process of Large Language Models

Zhanke Zhou[1 2 *] **Zhaocheng Zhu**[3 4 *] **Xuan Li**[1 *] **Mikhail Galkin**[6]
Xiao Feng[1] **Sanmi Koyejo**[2] **Jian Tang**[3 5] **Bo Han**[1 †]

Abstract

Numerous applications of large language models (LLMs) rely on their ability to perform step-by-step reasoning. However, the reasoning behavior of LLMs remains poorly understood, posing challenges to research, development, and safety. To address this gap, we introduce *landscape of thoughts* (LoT), the first landscape visualization tool to inspect the reasoning trajectories with certain reasoning methods on any multi-choice dataset. We represent the *textual states* in a trajectory as *numerical features* that quantify the states' distances to the answer choices. These features are then visualized in two-dimensional plots using t-SNE. Qualitative and quantitative analysis with the landscape of thoughts effectively distinguishes between strong and weak models, correct and incorrect answers, as well as different reasoning tasks. It also uncovers undesirable reasoning patterns, such as low consistency and high uncertainty. Additionally, users can adapt LoT to a model that predicts the property they observe. We showcase this advantage by adapting LoT to a lightweight verifier that evaluates the correctness of trajectories. Empirically, this verifier boosts the reasoning accuracy and the test-time scaling effect. The code is publicly available at https://github.com/tmlr-group/landscape-of-thoughts.

1 Introduction

Large language models (LLMs) have revolutionized the paradigm of solving problems. Many practical applications, *e.g.*, LLM as agent [41, 28, 64], critically depend on step-by-step reasoning [58, 27]. Despite progress in advanced models like OpenAI o1 [22] and decoding methods such as test-time scaling [44], the underlying *reasoning behavior* of LLMs remains poorly understood, hindering the development of these models and posing deployment risks [5].

A few pioneer attempts [53, 39, 40, 14] probe LLM reasoning, but their insights often hinge on specific decoders and tasks. In practice, practitioners debug by *manually reading* the reasoning trajectories generated by LLMs, which has two drawbacks: (i) **scalability**—human inspection does not scale (*e.g.*, at 30s per trajectory, 100 trajectories require 50min); and (ii) **aggregation**—deriving reliable, dataset-level conclusions (*e.g.*, from 10,000 trajectories) is difficult, yielding subjective and even biased summaries. These costs compound during iterative development, where fast, interpretable feedback is essential. Consequently, there is a clear need for general, reusable tools to analyze LLM reasoning in users' settings. This tool can potentially benefit *engineers* by speeding iteration, *reasoning researchers* by informing decoder improvements, and *safety researchers* by monitoring and improving model behavior.

To this end, we introduce the *landscape of thoughts* (LoT), a visualization of LLM reasoning trajectories that delivers automatic, objective analysis from single examples to full datasets. Analogous

*Equal Contribution, [1]TMLR Group, Hong Kong Baptist University, [2]Stanford University, [3]Mila - Québec AI Institute, [4]Université de Montréal, [5]HEC Montréal, [6]Intel AI Lab.
†Correspondence to Bo Han (bhanml@comp.hkbu.edu.hk).

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: ResponsibleFM.

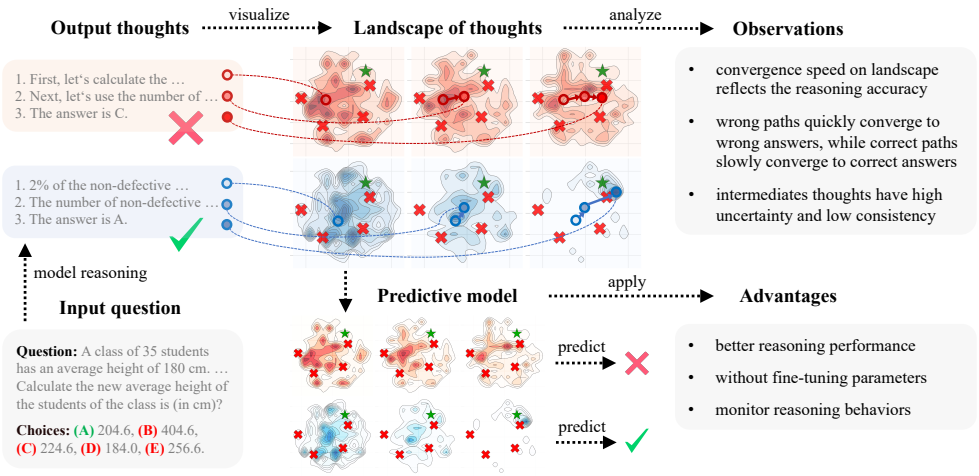


Figure 1: Landscape of thoughts for visualizing the reasoning steps of LLMs. Note that the red landscape represents wrong reasoning cases, while the blue indicates the correct ones. The darker regions in landscapes indicate more thoughts, with ✖ indicating incorrect answers and ★ marking correct answers. Specifically, given a question with multiple choices, we sample a few thoughts from an LLM and divide them into two categories based on correctness. We visualize the landscape of each category by projecting the thoughts into a two-dimensional feature space, where each density map reflects the distribution of states at a reasoning step. With these landscapes, users can easily discover the reasoning patterns of an LLM or a decoding method. In addition, a predictive model is applied to predict the correctness of landscapes and can help improve the accuracy of reasoning.

to the t-SNE [52], LoT highlights structure in high-dimensional reasoning space. By pairing qualitative landscapes with quantitative metrics (consistency, uncertainty, and perplexity), LoT enables comparison and reveals insights beyond manual inspection.

Specifically, given any multi-choice reasoning dataset, LoT visualizes the distribution of intermediate states in any reasoning trajectories of interest *w.r.t.* the answer choices, which enables users to uncover reasoning patterns in both success and failure trajectories (Fig. 1). The core idea is to characterize the *states of textual thoughts* in a trajectory as *numerical features* that quantify the states' distances to the answer choices. These distances are estimated by the perplexity metric, with the same LLM to generate thoughts and explain to itself. Then, these state features (i) produce three metric plots and (ii) are projected into a two-dimensional space with t-SNE to generate the landscape.

We examine LoT with different dimensions of model sizes, decoding methods, and reasoning datasets. LoT reveals several insightful observations regarding the reasoning behaviors of LLMs. Some notable observations include: 1) The convergence speed of trajectories towards correct answers reflects the accuracy, no matter what base model, decoding method, or dataset is used; 2) The convergence speed of trajectories in success and failure cases is distinct, indicating that we may use the convergence speed of a reasoning trajectory to predict its accuracy; 3) Low consistency and high uncertainty are generally observed in the intermediate thoughts, presenting the unstable properties of the reasoning process. LoT reveals them by bridging localized text understanding with global reasoning dynamics. To our knowledge, these patterns have not been reported by prior analyses of reasoning, which primarily rely on manual text inspection or aggregate performance metrics.

Since our tool is built on top of state features, it can be adapted to a machine-learning model to quantitatively predict certain properties, such as the findings mentioned above. We showcase this advantage by training a lightweight model to predict the success and failure cases, which is equivalent to verifiers commonly used in LLM reasoning [11]. Even though this verifier is lightweight compared to most LLM-based verifiers, it consistently improves the reasoning performance on most combinations of models, decoding methods, and datasets in our experiments. Hence, users can further leverage this advantage to predict the properties in their scenarios.

In summary, our main contributions are three-fold:

- We introduce the first tool for automatic and scalable visualization of the LLM reasoning procedure, applicable to any open-source models and decoding methods on multi-choice datasets (Sec. 2).

- Our tool reveals several observations regarding the reasoning behaviors of different language models, decoding methods, and reasoning datasets, offering several new insights (Sec. 3).
- Our tool can also be adapted to a model to predict certain properties and guide the reasoning process, improving LLM reasoning without modifying the model parameters (Sec. 4).

2 Landscape of Thoughts

2.1 Problem Formulation

Our goal is to *visualize* the reasoning trajectories of LLMs across a variety of task domains. Specifically, we target datasets consisting of *multiple-choice questions*, where each datapoint $(x, y, \mathcal{C})$ comprises a question x, a correct answer y, and a finite set of candidate choices $\mathcal{C} = \{c_j\}_{j=1}^k$, all represented in texts. [3] The visualization tool applies to the following models and methods.

Language models. To explore the landscape of thoughts generated by an LLM $p_{\text{LLM}}(\cdot)$, the model should produce diverse reasoning trajectories for solving a problem. In each trajectory, the reasoning thoughts are decoded autoregressively as $\hat{t}_i \sim p_{\text{LLM}}(t_i|x, \mathcal{C}, \hat{t}_1, \ldots, \hat{t}_{i-1})$: each thought $\hat{t}_i$ is conditioned on the question x, the candidate set $\mathcal{C}$, and the sequence of preceding thoughts $\hat{t}_1, \ldots, \hat{t}_{i-1}$. To characterize intermediate states within these trajectories, the LLM must also function as a likelihood estimator, enabling the computation of the probability $p_{\text{LLM}}(\hat{y}|x, \mathcal{C}, \hat{t}_1, \ldots, \hat{t}_i)$ of any answer $\hat{y}$. These two requirements are generally satisfied by open-source LLMs, such as Llama [13] and DeepSeek [32]. However, closed-source LLMs like GPT-4 [1] and Gemini [48] are excluded, as their likelihood estimation is not publicly supported.

Reasoning methods. While there are many approaches to solving reasoning problems with LLMs [12, 24], this work focuses on chain-of-thought (CoT) [58] and its derivatives [70, 63], owing to their widespread use and development. These decoding methods generally guide the model in generating a structured trajectory of intermediate reasoning thoughts before arriving at the final answer. To visualize a large number of reasoning thoughts effectively, these thoughts should be automatically parsed into distinct units (*e.g.*, via sentence tokenization). Most LLMs can satisfy this requirement. [4]

2.2 Qualitative Visualization with Landscapes

Given a collection of reasoning trajectories generated by an LLM, our tool seeks to visualize how different trajectories lead to either correct or incorrect answers within a two-dimensional (2D) space, as illustrated in Fig. 1. A key challenge lies in the absence of a direct mapping from the *textual* space of thoughts to *numerical* 2D coordinates. To address this gap, we utilize the same LLM to represent intermediate states as numerical features. These state features are then projected into a 2D space for visualization. For simplicity, we denote a thought as t_i instead of $\hat{t}_i$, which is clear in the following.

Characterizing the states. Here, the intermediate *thoughts* $\{t_i\}_{i=1}^n$ in a reasoning trajectory naturally define a sequence of *states* $\{s_i\}_{i=0}^n$, where $s_0 = [x]$ and $s_i = [x, t_1, t_2, \ldots, t_i]$. Here, we propose to characterize the states as features using the likelihood function of the LLM. Specifically, the k-dim feature $\boldsymbol{f}_i$ for state s_i indicates the relative distances from the state s_i to all possible choices $\{c_j\}_{j=1}^k$:

$$\boldsymbol{f}_i \triangleq [d(s_i, c_1), d(s_i, c_2), \ldots, d(s_i, c_k)]^\top, \tag{1}$$

where $d(s_i, c_j)$ measures the *distance* between state s_i and choice c_j. To reduce the effect of length on choices, we calculate $d(s_i, c_j)$ through the perplexity metric [42, 35]:

$$d(s_i, c_j) \triangleq \exp\left(-\frac{1}{|c_j|}\sum_{t=1}^{|c_j|}\log p_{\text{LLM}}(c_j[t]|s_i, c_j[:t])\right) = p_{\text{LLM}}(c_j|s_i)^{-1/|c_j|}, \tag{2}$$

where $|c_j|$ is the number of tokens in c_j, and $p_{\text{LLM}}(c_j|s_i)$ is the accumulated probability in an autoregressive manner. Assume $|c_j| = T$, we have $p_{\text{LLM}}(c_j|s_i) = p_{\text{LLM}}(c_j[1]|s_i) \cdot p_{\text{LLM}}(c_j[2]|s_i, c_j[1]) \cdot p_{\text{LLM}}(c_j[3]|s_i, c_j[1], c_j[2]) \ldots p_{\text{LLM}}(c_j[T]|s_i, c_j[1], c_j[2] \ldots c_j[T-1])$. The token-level probabilities are normalized over the entire vocabulary; $c_j[1]$ is the first token of c_j, and $c_j[T]$ is the last token.

[3] LoT is positioned for multi-choice questions. Appendix B.7 discusses its extension to open-ended tasks.

[4] We empirically verify the robustness of LoT if this requirement does not hold (please see Appendix E.9).

We further normalize the vector $\boldsymbol{f}_i$ to have a unit ℓ_1 normalization. Additionally, to represent the choices as landmarks in the visualization, it is necessary to encode the choices as feature vectors. Notably, the perplexity decreases as the model's prediction confidence increases. To align with this observation, we define the feature vector $\boldsymbol{f}_j^c$ for a choice c_j as:

$$\boldsymbol{f}_j^c \triangleq \frac{1}{k}[\mathbb{1}(j \neq 1), \ldots, \mathbb{1}(j \neq k)]^\top. \tag{3}$$

For r trajectories, each with n states, we compute the feature vectors for all $r \cdot n$ states. The indicator function $\mathbb{1}(j \neq 1)$ will output 1 if j is not 1, and output 0 if and only if j is 1. Here, the element 0 in c_1 indicates this choice has 0 distance to the first anchor, which is c_1 itself. Besides, elements 1 indicate that the distances among anchors are assumed to be the same. [5] Together with the feature vectors of k choices, we obtain a feature matrix $\boldsymbol{F} \in \mathbb{R}^{k \times (r \cdot n + k)}$ as:

$$\boldsymbol{F} \triangleq [\boldsymbol{f}_1^{(1)}, \ldots, \boldsymbol{f}_n^{(1)}, \ldots, \boldsymbol{f}_1^{(r)}, \ldots, \boldsymbol{f}_n^{(r)}, \boldsymbol{f}_1^c, \ldots, \boldsymbol{f}_k^c]. \tag{4}$$

Note that a sufficiently large number of trajectories is necessary to generate a comprehensive visualization of the reasoning landscape. For computational efficiency, we sample d trajectories per question across all questions, yielding $r = d|\text{questions}|$ total trajectories. We then normalize feature vectors by reordering choices so the correct answer appears in the first dimension across all questions. As such, we can visualize the landscape of multiple questions by putting their trajectories together, which is more efficient than visualizing by generating enough trajectories per question.

Visualization. After constructing the feature matrix $\boldsymbol{F}$, we project the states and choices into a 2D space for visualization. This step can be accomplished using existing methods of dimensionality reduction [38, 52, 36]. We employ t-SNE [52] due to its ability to preserve the underlying manifolds of the original high-dimensional space and its robustness to a wide range of transformations. [6] By applying t-SNE to the k-dim $\boldsymbol{F}$, we obtain the 2-dim coordinates $\bar{\boldsymbol{F}} \in \mathbb{R}^{2 \times (rn+k)}$. The two dimensions are reduced from the original space, which represents all possible answers, and each state's coordinates show its distance from different answers. Finally, the coordinates of the states define a discrete density function in the 2D space, presented by the landscape's color depth.

2.3 Quantitative Visualization with Metrics

Besides the qualitative visualization, we introduce three quantitative metrics to help understand the LLMs' behavior. These metrics are defined based on the intermediate states in Sec. 2.2.

Consistency. To understand whether the LLM knows the answer before generating all thoughts, we compute the consistency of state s_i by checking whether $\boldsymbol{f}_i$ and $\boldsymbol{f}_n$ agree

$$\text{Consistency}(s_i) = \mathbb{1}(\arg\min \boldsymbol{f}_i = \arg\min \boldsymbol{f}_n). \tag{5}$$

Uncertainty. To know how confident the LLM is about its predictions at intermediate steps, we compute the uncertainty of state s_i as the entropy of $\boldsymbol{f}_i$ (note $\sum_{d \in \boldsymbol{f}_i} d = 1$)

$$\text{Uncertainty}(s_i) = -\sum_{d \in \boldsymbol{f}_i} d \cdot \log d. \tag{6}$$

Perplexity. We are also interested in how confident the LLM is about its thoughts. We use the perplexity of thought t_i, since it is comparable across thoughts of different length

$$\text{Perplexity}(t_i) = p_{\text{LLM}}(t_i | s_{i-1})^{-1/|t_i|}. \tag{7}$$

Remark 2.1. Note that in previous works, these metrics are mainly used to evaluate the performance of language modeling on each token. We repurpose them to analyze intermediate thoughts in the trajectories, which is a new lesson for the community. Appendix C introduces related works in detail. The following section demonstrates that the LoT, containing the qualitative landscape and the quantitative metrics, is effective for automatic and scalable visualization of reasoning trajectories.

[5] LoT can be applied to trajectories with different numbers of states. We assume n states for demonstrations.

[6] Appendix E.8 shows that LoT is compatible and robust with different methods of dimensionality reduction.

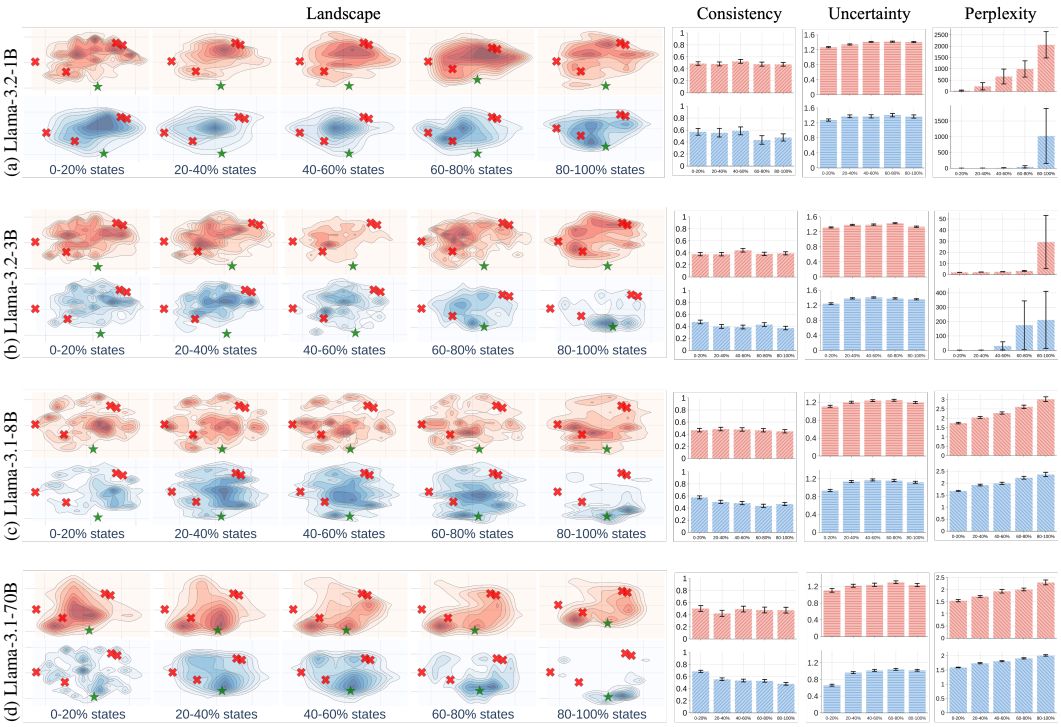


Figure 2: Comparing the LoT of different language models (with CoT on the AQuA dataset). Darker regions represent higher state density, with ✖ indicating incorrect answers and ★ marking the correct ones. Through the reasoning trajectories, spanning from early (0-20% states) to the later stages (80-100% states), the visualization shows correct cases (bottom rows in blue) with incorrect cases (top rows in red). Metrics are calculated *w.r.t.* each bin, *e.g.*, 20% - 40% of states. The reasoning accuracy of the four subfigures is: (a) 15.8%, (b) 42.0%, (c) 53.2%, and (d) 84.4%.

3 Results and Observations

In this section, we utilize the landscape of thoughts to analyze the reasoning behavior of LLMs by comparing the visualizations across three dimensions: (1) diverse scales and types of *language models* in Sec. 3.1, (2) different *reasoning tasks* in Sec. 3.2, and (3) various *reasoning methods* in Sec. 3.3. Unless stated otherwise, we employ Llama-3.1-70B with CoT as the default configuration in evaluations. All the visualizations are built upon the model's estimation of their own thoughts. [7]

3.1 Comparison across Language Models

We study several LLMs' behavior across parameter scales (from 1B, 3B to 70B). We run each model with CoT prompting on 50 randomly selected problems from the mathematical reasoning dataset AQuA. Their landscapes are shown in Fig. 2, from which we have the following observations.

Observation 3.1 (*The landscape converges faster as the model size increase*)**.** As model parameters scale from 1B to 70B, the corresponding landscape demonstrates faster convergence to the correct answers with higher density in the last 20% states, aligning with the increasing accuracy. With more parameters, larger models can store broader knowledge [3]. This leads to more confident solutions, demonstrated by more focused answer patterns and lower uncertainty.

Observation 3.2 (*Larger models have higher consistency, lower uncertainty, and lower perplexity*)**.** As the model size increases, the consistency increases; at the same time, the uncertainty and perplexity decrease significantly. This also aligns with the higher accuracy for the large models. [8]

In addition, we apply LoT to up-to-date reasoning models QwQ 32B [49] and observe:

[7] Appendix E.1 validates each qualitative observation from LoT. Full visualizations are in Appendix F.

[8] Appendix E.3 presents additional analyses of the consistency metric: the consistency does not relate to the length of the trajectory. In addition, Appendix E.5 supports the validity of comparing perplexity across models.

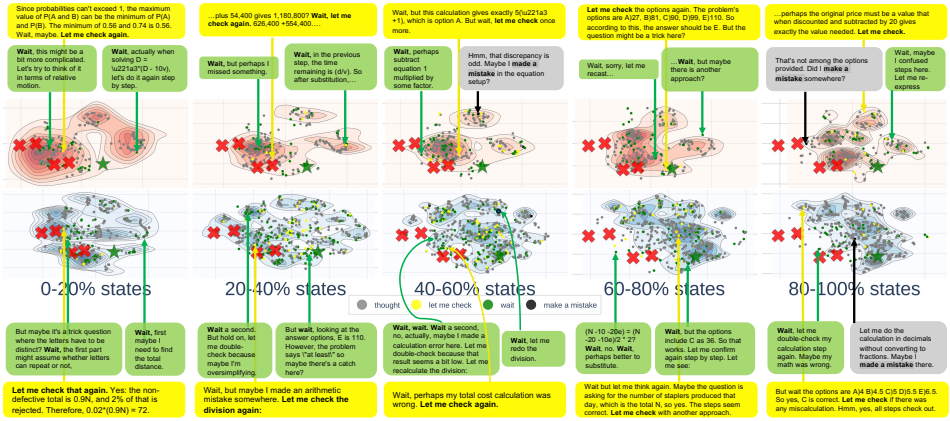


Figure 3: The LoT of the reasoning model QwQ-32B (using CoT prompting on the AQuA dataset).

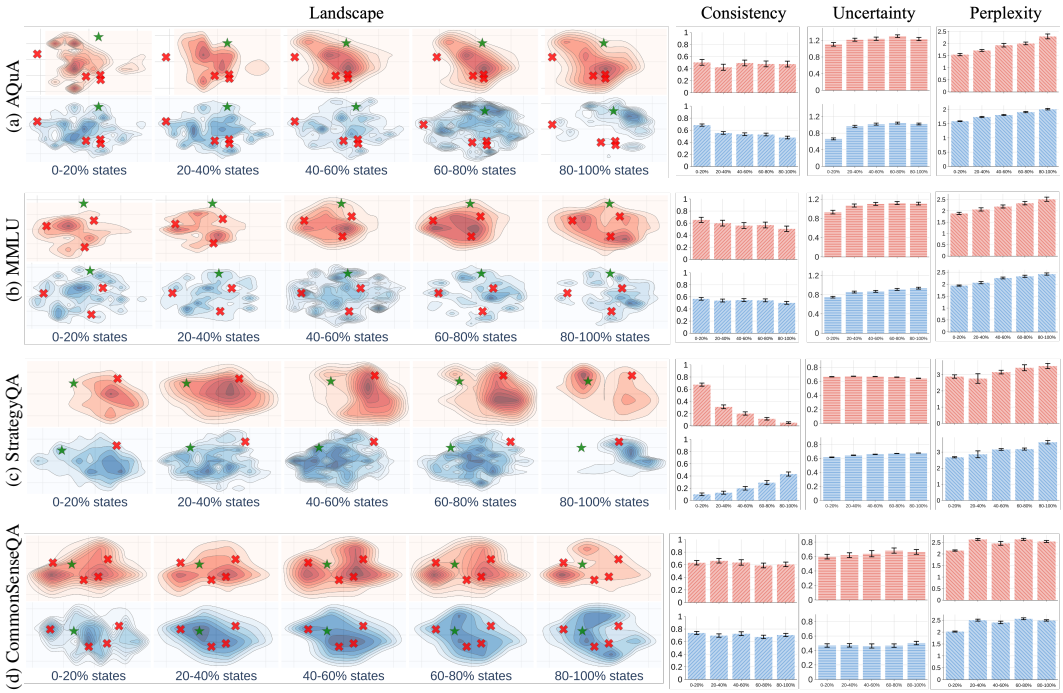


Figure 4: Comparing the LoT of different datasets (using Llama-3.1-70B with CoT). The accuracy of reasoning for the four subfigures is: (a) 84.4%, (b) 80.2%, (c) 75.8%, and (d) 64.8%.

Observation 3.3 (*Reasoning models present more-complex reasoning behaviors in landscapes.*)**.** As shown in Fig. 3, the landscapes can capture complex reasoning patterns such as self-evaluation and self-correction. Specifically, correct trajectories tend to include more instances of self-evaluation and self-correction compared to incorrect ones. These behaviors often occur early in the reasoning process, especially when the model is far from the correct one. Compared to non-reasoning models, correct trajectories here show greater diversity, with green and yellow points more widely scattered.

3.2 Comparison across Reasoning Tasks

Besides AQuA dataset, we include MMLU, CommonsenseQA, and StrategyQA datasets. We run the default model with CoT on 50 problems per dataset. These observations are derived from Fig. 4:

Observation 3.4 (*Similar reasoning tasks exhibit similar landscapes*)**.** The landscapes of AQuA, MMLU, and StrategyQA in Fig. 4 exhibit organized search behavior with higher state diversity, while CommonSenseQA presents concentrated search regions, reflecting direct retrieval of common-

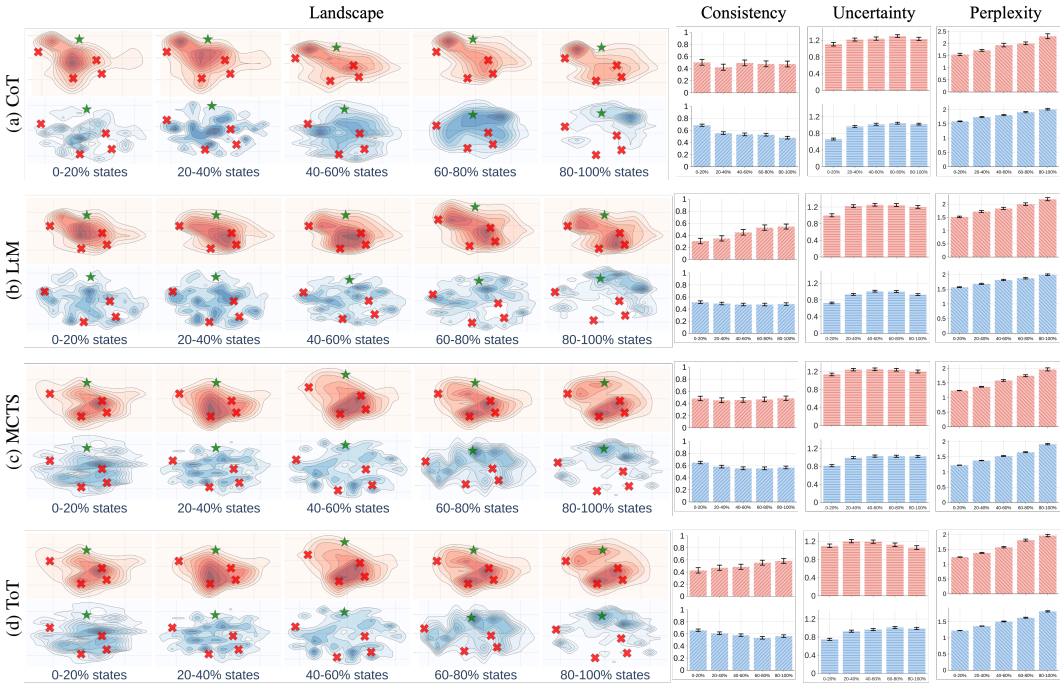


Figure 5: Comparing the LoT of four reasoning methods (using Llama-3.1-70B on the AQuA dataset). The reasoning accuracy is: (a) 84.4%, (b) 82.2%, (c) 75.8%, and (d) 81.6%, respectively.

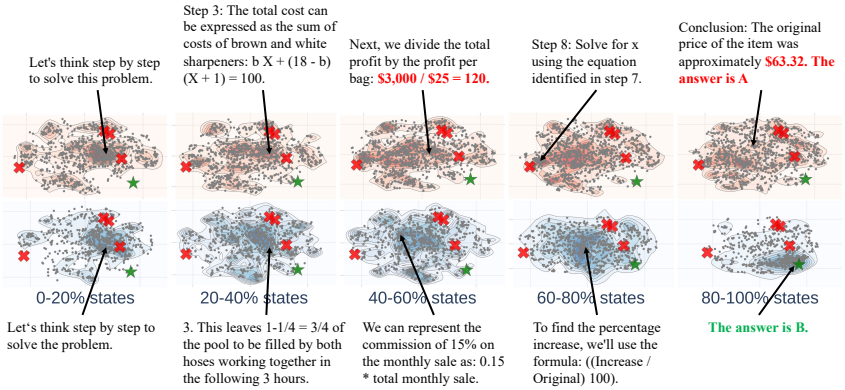


Figure 6: Case Study of LoT, with Llama-3.1-8B using CoT on AQuA.

sense knowledge rather than step-by-step reasoning processes. These distinct landscape patterns demonstrate the potential to reveal underlying domain relationships across different reasoning tasks.

Observation 3.5 (*Different reasoning tasks present significantly different patterns in consistency, uncertainty, and perplexity*)**.** The histograms in Fig. 4 show that the perplexity consistently increases as reasoning progresses across all datasets. Specifically, different datasets, *e.g.*, AQuA and MMLU, show distinctly higher levels of uncertainty. As for StrategyQA, correct trajectories show increasing consistency that surpasses incorrect trajectories at around 60% states, while incorrect trajectories show decreasing consistency. However, when the trajectory is longer than the ground truth trajectory, the later stages (60-100% of states) exhibit both increasing perplexity and decreasing uncertainty. [9]

3.3 Comparison across Reasoning Methods

Setup. We evaluate the default model with four reasoning methods: chain-of-thought (CoT) [58], least-to-most (LtM) [70], MCTS [68], and tree-of-thought (ToT) [63]. We run these methods on 50 problems from AQuA and observe that:

[9] We show detailed analysis for trajectories in StrategyQA in Appendix E.4.

Observation 3.6 (*Cross-method comparison: Among correct reasoning trajectories, methods with faster convergence to correct answers achieve higher accuracy.*)**.** From Fig. 5, we observe that the states scatter dispersedly at early stages and gradually converge to correct (or incorrect) answers in later stages. Here, converge means the trend of a reasoning trajectory approaching one answer. Generally, methods with more scattered landscapes (that converge more slowly) present lower accuracy than those that converge faster. For example, the blue landscape in Fig. 5(a) converges faster than the blue landscapes in Fig. 5(c), and the former is with a higher accuracy than the latter.

Observation 3.7 (*Within-method comparison: For any single method, incorrect trajectories converge faster to wrong answers than correct trajectories converge to right answers.*)**.** As can be seen from Fig. 4, failure trajectories usually converge to the wrong answers at earlier states of reasoning, e.g., 20-40% states in Fig. 4(c). By contrast, the states in the success trajectories converge to the correct answers at later 80-100% states. This implies that early states of the reasoning process can lead to any potential answers (from a model perspective), while the correct answers are usually determined at the end of reasoning trajectories. In addition, Fig. 6 showcases the corresponding text of thoughts. [10]

Observation 3.8 (*Compared to failure trajectories, the intermediate states in correct trajectories have higher consistency w.r.t. the final state*)**.** By comparing the consistency plots in Fig. 5, we found that the model generally has low consistency between the intermediate states and the final state. Notably, the consistency of wrong trajectories is significantly lower than that of correct trajectories. This implies that the reasoning process can be quite unstable. Even though decoding methods like CoT and LtM are designed to solve a problem directly (without exploration), the generated thoughts by these methods do not consistently guide the reasoning trajectory to the answer.

4 Adapting Visualization to Predictive Models

One advantage of our method is that it can be adapted to a model to predict any property users observe. Here, we show how to convert our method to a lightweight verifier for voting trajectories, following the observations in Sec. 3. Note that this methodology is not limited to verifiers. Users can use this technique to adapt the visualization tool to monitor the properties in their scenarios.

4.1 A Lightweight Verifier

Observation 3.7 and 3.8 show that the convergence speed and consistency of intermediate states can distinguish correct and wrong trajectories. Inspired by these observations, we build a model $g : \mathbb{R}^{(k+1)\times n} \to \{0, 1\}$ to predict the correctness of a trajectory based on the state features $\{\boldsymbol{f}_i\}_{i=1}^n$ and consistency metric $\{\text{Consistency}(\boldsymbol{f}_i)\}_{i=1}^n$. The insight is that the state features, used to compute the 2-D visualization, encode rich location information of the states and can be used to estimate the convergence speed. Due to the small dimensionality of these features, we parameterize f with a random forest [6] to avoid overfitting. We use this model as a verifier to enhance LLM reasoning [11]. Unlike popular verifiers [30] that involve a moderately sized language model on textual thoughts, our verifier operates on state features and is quite lightweight. We train a verifier on thoughts sampled on the training split of each dataset and apply it to vote trajectories at test time. Given q trajectories sampled by a decoding method, the final prediction is produced by a weighted majority voting:

$$\hat{y} = \arg\max_{c \in \mathcal{C}} \sum_{i=1}^{q} \quad \mathbb{1}(\hat{y}^{(i)} = c) \cdot g(\{\boldsymbol{f}_i\}_{i=1}^n, \{\text{Consistency}(\boldsymbol{s}_i)\}_{i=1}^n). \tag{8}$$

4.2 Experimental Results

We evaluate our numerical verifier against an unweighted voting baseline [55] with various models, decoding methods, and reasoning datasets. We report the accuracy here instead of commonly seen pass@k, which will be easily hacked by a simple random guess or a traverse of all candidates to obtain a high score. Detailed settings of experiments are in Appendix D. We also provide ablation studies on training the verifier and discuss and compare the variance of the verifier in Appendix E.6, and experiment on the scaling effect with different features in Appendix E.7.

Effectiveness of the verifier. We first compare our verifier against the unweighted voting baseline, each applied to 10 trajectories. As shown in Fig. 7, our verifier consistently enhances the reasoning performance of all models and decoding methods, even though our verifier does not use any pre-trained language model. Notably, smaller language models (1B and 3B) show significant performance

[10] In Appendix E.2, only a few incorrect trajectories (1.8%) are close to the correct answer in middle thoughts.

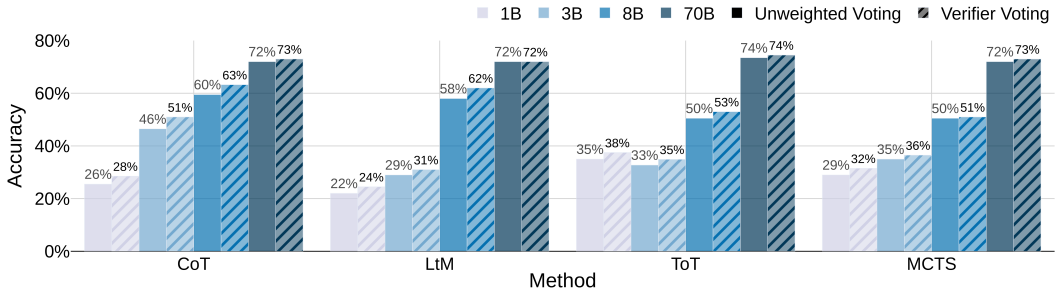


Figure 7: The accuracy of reasoning under different decoding methods and model scales (averaging across all four datasets). Results for each dataset are in Appendix F.

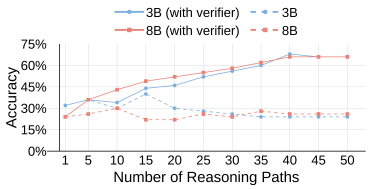
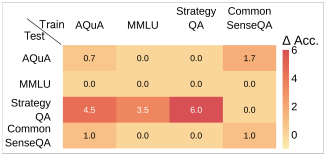


Figure 8: Demonstration of the inference-time scaling effect of the verifier. We show the voting accuracy (%) on StrategyQA scales with the number of trajectories.



(a) Transfer across datasets

(b) Transfer across models

Figure 9: Absolute accuracy changes (Δ Acc) with the verifier, compared to performance in Fig. 7 (without the verifier). The verifier is trained on each column (dataset or model) and evaluated on all rows (other datasets or models). Positive values indicate improvement in accuracy with the verifier.

gains with the verifier's assistance, achieving substantial improvements over their original capabilities of reasoning. We also compare the verifiers between reward-guided methods.

Test-time scaling. While the improvement of the verifier seems marginal with 10 trajectories, our verifier can provide a substantial performance gain with more trajectories. We adjust the number of trajectories from 1 to 50, and plot the results of the verifier and the unweighted voting baseline in Fig. 8. Models with our verifier exhibit significantly stronger scaling behaviors, achieving over 65% accuracy. In contrast, the performance of the baseline saturated around 30% accuracy. These results suggest that our state features, which are used in both the visualization tool and the verifier, capture important information about the reasoning behavior of LLMs. Thus, the verifier can boost test-time scaling, especially in solving complex problems.

Cross-dataset and cross-model transferability. One interesting property of the state features and metrics is that their shape and range are agnostic to the model and dataset, suggesting that we may deploy the verifier trained on one dataset or model in another setting. As illustrated in Fig. 9, we evaluate how the verifier transfers across reasoning datasets (*e.g.*, train on AQuA and test on MMLU) and model scales (*e.g.*, train on 1B model and test on 70B model). We observe some positive transfers across datasets and models. For example, a verifier trained on AQuA can improve the performance of StrategyQA by 4.5%. A verifier trained on the 70B model also improves the performance of the 3B model by 5.5%. However, some cases do not benefit from the transferring verifiers. We leave improving the transferability of the state features and metrics as future work.

5 Conclusion

This paper introduces the landscape of thoughts, a visualization tool for analyzing the reasoning trajectories produced by large language models with chain-of-thought. Built on top of feature vectors of intermediate states in trajectories, our tool reveals several insights into LLM reasoning, such as the relationship between convergence and accuracy, and issues of low consistency and high uncertainty. Our tool can also be adapted to predict the answer of reasoning trajectories based on the observed property, which is demonstrated by a lightweight verifier developed based on the feature vectors and our observations for distinguishing the correctness of trajectories. We foresee that this tool will create several opportunities to develop, understand, and monitor the LLM reasoning.

References

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

[2] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

[3] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*, 2024.

[4] Ehsan Amid and Manfred K Warmuth. Trimap: Large-scale dimensionality reduction using triplets. *arXiv preprint arXiv:1910.00204*, 2019.

[5] Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, et al. Foundational challenges in assuring alignment and safety of large language models. *TMLR*, 2024.

[6] Leo Breiman. Random forests. *Machine learning*, 2001.

[7] T Tony Cai and Rong Ma. Theoretical foundations of t-sne for visualizing high-dimensional clustered data. *Journal of Machine Learning Research*, 23(301):1–54, 2022.

[8] Qiguang Chen, Libo Qin, Jiaqi Wang, Jinxuan Zhou, and Wanxiang Che. Unlocking the capabilities of thought: A reasoning boundary framework to quantify and optimize chain-of-thought. In *NeurIPS*, 2024.

[9] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. In *ICLR*, 2024.

[10] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does bert look at? an analysis of bert's attention. In *ACL Workshop BlackboxNLP*, 2019.

[11] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[12] Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

[13] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[14] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. In *NeurIPS*, 2024.

[15] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

[16] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *TACL*, 2021.

[17] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. In *arXiv*, 2025.

[18] Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *EMNLP*, 2023.

[19] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021.

[20] John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *EMNLP*, 2019.

[21] Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A Choquette-Choo, and Nicholas Carlini. Preventing verbatim memorization in language models gives a false sense of privacy. *arXiv preprint arXiv:2210.17546*, 2022.

[22] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[23] Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. The impact of reasoning step length on large language models. *arXiv preprint arXiv:2401.04925*, 2024.

[24] Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran. Lambada: Backward chaining for automated reasoning in natural language. In *ACL*, 2023.

[25] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *ICLR*, 2023.

[26] Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Attention is not only a weight: Analyzing transformers with vector norms. In *EMNLP*, 2020.

[27] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.

[28] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*, 2020.

[29] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *NeurIPS*, 2023.

[30] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[31] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *ACL*, 2017.

[32] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[33] Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, et al. Improve mathematical reasoning in language models by automated process supervision. In *arXiv*, 2024.

[34] Aman Madaan and Amir Yazdanbakhsh. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*, 2022.

[35] C. Manning. *Foundations of statistical natural language processing*. The MIT Press, 1999.

[36] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[37] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[38] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 1901.

[39] Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *ICLR*, 2023.

[40] Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi, Najoung Kim, and He He. Testing the general deductive reasoning capacity of large language models using ood examples. In *NeurIPS*, 2023.

[41] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023.

[42] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 1948.

[43] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *ICML*, 2023.

[44] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

[45] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 2017.

[46] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL*, 2019.

[47] Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. Large language models are in-context semantic reasoners rather than symbolic reasoners. *arXiv preprint arXiv:2305.14825*, 2023.

[48] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[49] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

[50] Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *ACL*, 2019.

[51] Jean-Francois Ton, Muhammad Faaiz Taufiq, and Yang Liu. Understanding chain-of-thought in llms through information theory. *arXiv preprint arXiv:2411.11984*, 2024.

[52] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[53] Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *ACL*, 2023.

[54] Xinyi Wang, Alfonso Amayuelas, Kexun Zhang, Liangming Pan, Wenhu Chen, and William Yang Wang. Understanding the reasoning ability of language models from the perspective of reasoning paths aggregation. In *ICML*, 2024.

[55] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023.

[56] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73, 2021.

[57] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *NeurIPS*, 2024.

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

[59] Skyler Wu, Eric Meng Shen, Charumathi Badrinath, Jiaqi Ma, and Himabindu Lakkaraju. Analyzing chain-of-thought prompting in large language models via gradient-based feature attributions. *arXiv preprint arXiv:2307.13339*, 2023.

[60] Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *ICLR*, 2024.

[61] Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. In *arXiv*, 2025.

[62] Tianyun Yang, Ziniu Li, Juan Cao, and Chang Xu. Mitigating hallucination in large vision-language models via modular attribution and intervention. In *ICLR*, 2025.

[63] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023.

[64] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

[65] Yunzhi Yao, Ningyu Zhang, Zekun Xi, Mengru Wang, Ziwen Xu, Shumin Deng, and Huajun Chen. Knowledge circuits in pretrained transformers. In *NeurIPS*, 2024.

[66] Xi Ye, Srinivasan Iyer, Asli Celikyilmaz, Ves Stoyanov, Greg Durrett, and Ramakanth Pasunuru. Complementary explanations for effective in-context learning. *arXiv preprint arXiv:2211.13892*, 2022.

[67] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. In *NeurIPS*, 2022.

[68] Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. In *NeurIPS*, 2024.

[69] Tianhang Zhang, Lin Qiu, Qipeng Guo, Cheng Deng, Yue Zhang, Zheng Zhang, Chenghu Zhou, Xinbing Wang, and Luoyi Fu. Enhancing uncertainty-based hallucination detection with stronger focus. In *EMNLP*, 2023.

[70] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*, 2023.

[71] Zhaocheng Zhu, Yuan Xue, Xinyun Chen, Denny Zhou, Jian Tang, Dale Schuurmans, and Hanjun Dai. Large language models can learn rules. *arXiv preprint arXiv:2310.07064*, 2023.

Appendix

A Impact Statement

Our work presents a tool for visualizing and understanding reasoning steps in large language models. We foresee that our work will introduce more interpretability and transparency into the development and deployment of LLMs, advancing us toward more trustworthy machine learning. However, we must acknowledge that malicious activities can also be augmented by our tool. For example, attackers may use this tool to find prompts that bypass the alignment safeguards in LLMs. We believe such risks will be mitigated if this tool is widely adopted by safety researchers. Overall, the positive societal consequences of our work outweigh the negative ones, which stem primarily from misuse.

B Further Discussions

B.1 Challenges in Analyzing LLM's Reasoning Automatically

Currently, the fundamental mechanisms behind both successful and unsuccessful reasoning attempts in LLMs remain inadequately understood. Traditional performance metrics, such as accuracy, provide insufficient insights into model behavior. While human evaluation has been employed to assess the quality of sequential thoughts (e.g., logical correctness and coherence), such approaches are resource-intensive and difficult to scale. We identify three challenges in developing *automated analysis systems* for LLMs' reasoning:

Challenge 1: Bridging the token-thought gap. Current explanatory tools, including attention maps [10, 26], probing [2, 50, 20], and circuits [15, 65], primarily operate at the token-level explanation. While these approaches offer valuable insights into model inference, they struggle to capture the emergence of higher-level reasoning patterns from lower-level token interactions. Additionally, the discrete nature of natural language thoughts poses challenges for traditional statistical analysis tools designed for continuous spaces. Understanding how thought-level patterns contribute to complex reasoning capabilities requires new analytical frameworks that can bridge this conceptual gap.

Challenge 2: Analyzing without training data access. Existing investigations into LM reasoning have predominantly focused on correlating test questions with training data [21, 54]. This approach becomes particularly infeasible given the reality of modern LLMs: many models are closed-source, while some offer only model weights. Therefore, a desired analysis framework should operate across varying levels of model accessibility.

Challenge 3: Measuring reasoning quality. Beyond simple performance metrics, we need new ways to evaluate the quality and reliability of model reasoning. This includes developing techniques to understand reasoning paths, creating intermediate representations that capture both token-level and thought-level patterns, and designing metrics that can assess the logical coherence and validity of reasoning steps.

Consequently, we propose that a viable analysis of reasoning behavior should satisfy multiple criteria: it should operate in a post-hoc manner with varying levels of model access, bridge the gap between token-level and thought-level analysis, and provide meaningful metrics for evaluating reasoning quality. Given the absence of tools meeting these requirements, we identify the need for a new analytical framework that can address these challenges while providing useful insights for improving model reasoning capabilities.

B.2 A Comparison Between Landscape Visualization and Textual Analysis

Notably, for the language model, one could manually examine the responses to individual questions, as their responses are interpretable by humans. However, this approach has two major limitations:

Limitation 1: Lack of Scalability. Analyzing the individual question is time-consuming and labor-intensive. In general, text-based analysis requires human evaluators to carefully read long reasoning chains word by word. For example, if it takes 30 seconds to understand a single problem, reviewing 100 problems would require around 50 minutes of focused human effort. This burden grows quickly, especially as researchers often repeat this process many times while developing models and methods. In practice, researchers need quick, easily interpretable feedback, such as accuracy, when experimenting with changes to models and methods.

Limitation 2: Lack of Aggregation. It is difficult to aggregate insights across multiple problems to understand model behavior at the dataset level. Summarizing model behavior across multiple problems presents another challenge. Suppose one researcher has 100 reasoning chains; it is hard for him/her to reliably synthesize the model's overall behavior. Different researchers may arrive at different, subjective summaries, which hinders consistency and interpretability.

By contrast, our visualization method provides a more objective and automatic way to analyze a model, making it much easier for researchers to analyze the model's reasoning behavior. Similar to the t-SNE [52], the visualization enables a more comprehensive analysis of multiple reasoning problems instead of only one problem. The visualization uniquely combines human-readable paths with quantitative, scalable metrics for reasoning process analysis, enabling both model comparisons and mechanistic insights beyond manual text inspection.

Notably, the landscape provides unique insights into LLM reasoning that text analysis alone cannot capture. This power source bridges the gap between localized text understanding and global reasoning behavior. Our analysis in Sec. 3 reveals insights that are not revealed by previous text-based analysis. These insights include structural patterns across many reasoning paths, a strong correlation between early consistency and accuracy, and model-level differences where larger models explore more broadly than smaller ones.

B.3 The Intrinsic Relationship Between Visualization and Metrics

In the modeling of this work, we project each thought (state in a trajectory) from text space to numerical space, with the thought's feature vector that each dimension indicates the distance to a particular answer (see Eqn. 1). We compute the feature vectors of all the thoughts from multiple trajectories and then obtain the feature matrix $\boldsymbol{F}$. Then, based on this feature matrix, we compute (1) the landscape visualization through dimension reduction and (2) the metrics of consistency and uncertainty. From this view, the metrics' information can actually be seen from the landscape. In this work, we mainly focus on the landscapes and also use the metrics plots to help analyze.

In addition, landscape visualizations preserve the information of metrics, including the consistency, uncertainty, convergence, and many other metrics that are not covered in this work. The landscape provides a "global" view of the overall reasoning trajectories, while each metric provides a "local" view of a particular aspect. Note that humans naturally prefer visual matters like figures and videos, e.g., researchers prefer to use t-SNE in understanding the classification models. We recommend using landscape as a visualization tool to help understand the LLM reasoning, while the metric plots can further help inspect some particular aspects.

B.4 Discussion on Results and Observations

In the landscape visualizations, red regions map out the reasoning trajectories that end in incorrect answers, while blue regions map out those that end correctly. The contour lines and the depth of color together convey the density of reasoning states at each step: darker shades mean more trajectories passing through that region. As you observe a landscape evolve from its initial scatter of states toward later clustering, you're seeing whether and how quickly the model's reasoning paths lock onto an answer.

Observation 3.6 arises when we compare only the blue (correct) landscapes of different methods in Fig. 5. Early in the process, all methods scatter widely, exploring many possibilities; over time, though, some methods' contours tighten more rapidly than others. Here, the landscape in Fig. 5a converges to its correct region much sooner—and with a denser cluster—than the landscapes in Fig. 5b to 5c, and this faster, tighter convergence corresponds to its higher accuracy. Namely, methods with more scattered landscapes (converge more slowly) present lower accuracy than those that converge faster.

A related pattern appears when we compare models of different sizes in Fig. 2 (Observation 3.1). As we scale from the 1B model to the 70B model, the last 20% of the reasoning steps show increasingly dense blue clusters. Larger models, with greater capacity to store and retrieve information, steer their reasoning more directly and confidently toward the right answer, mirroring their higher accuracy. This further supports the positive correlation between convergence speed (of correct landscapes) and reasoning accuracy, which is revealed in Observation 3.6.

Observation 3.7 emerges from contrasting the red and blue landscapes of the same algorithm in Fig. 5. Here, failure trajectories (red) often settle into a wrong answer by roughly 20-40% of the reasoning process, while success trajectories (blue) only coalesce around the correct answer toward the very end—around 80-100% of the states. This indicates that early reasoning states are exploratory and can drift toward incorrect conclusions, whereas correct solutions only converge late in the trajectories. This convergence-speed disparity between red and blue landscapes also holds across multiple datasets in Fig. 4.

Finally, Fig. 4 shows that each reasoning task leaves a distinct landscape "fingerprint," supporting Observation 3.4. In AQuA, MMLU, and StrategyQA, the landscapes trace wide, structured sweeps of reasoning states—clear evidence of step-by-step deduction and exploration of intermediate hypotheses. By contrast, CommonSenseQA produces a tightly clustered trajectory from the outset, indicating direct retrieval of knowledge rather than an iterative trajectory. This divergence mirrors the tasks themselves: AQuA, MMLU, and StrategyQA require exploratory traversal through multiple reasoning steps, resulting in diverse yet organized state distributions, whereas CommonSenseQA depends on straightforward recall. These task-specific structures demonstrate how our landscape visualizations can uncover both shared patterns and fundamental differences across reasoning challenges.

In addition, each of these qualitative observations is further supported by statistical analyses in Appendix E.1, and we provide full visualizations, including annotated state trajectories (Figs. 23 to 26) and additional model comparisons (Figs. 27 to 28).

B.5 Potential Extension to Pruning Unpromising Trajectories

We showcase that our tool can be utilized to identify potentially incorrect reasoning trajectories at test time. In Section 3.3, we build up a lightweight verifier, which is based on the thoughts' feature vectors and the consistency metric from the landscape of thoughts. This verifier indeed aims to predict the correctness of a reasoning trajectory, in order to boost the reasoning accuracy at test time. It is proven to be beneficial to the voting of multiple reasoning trajectories, as shown in Sec. 4.2.

Further, this verifier (together with the visualization tool) can be adopted to prune unpromising reasoning trajectories in tree-based searching. For instance, in methods like tree-of-thoughts and MCTS, a model explores multiple reasoning trajectories and usually uses the same model to identify the promising paths to search for the ultimate solution. Here, by leveraging features from the landscape of thoughts and the consistency metric, our verifier can identify flawed trajectories early during reasoning, acting as an efficient pruning mechanism to boost the search efficiency and reasoning performance.

Therefore, our tool can be integrated into the reasoning methods to monitor particular reasoning patterns (e.g., the correctness) and help understand as well as boost reasoning. There are multiple directions that deserve future exploration, including the one to identify and prune the potentially incorrect reasoning trajectories.

B.6 Potential Extension to Identify Post-hoc Trajectories

In the following, we discuss the feasibility of detecting post-hoc trajectory using our framework, particularly in defining the post-hoc trajectory. A post-hoc trajectory refers to the trajectory that the model exhibits high confidence in a single answer in the early states and maintains high consistency across states in the trajectory. Specifically,

- the "early state" correspond to the "very early tokens of the response";
- the "high confidence in a single answer" corresponds to the "model has chosen its answer";
- the "high consistency across states in the trajectory" corresponds to the "trajectory is produced as a consequence of that decision".

Namely, the post-hoc trajectory can be potentially identified by inspecting the confidence and consistency of particular positions of states in our framework. Then, we elaborate on the more detailed definitions for the three components above.

- For defining the "early states", it should have an absolute threshold of states index, e.g., early 10 states, or a relative threshold, e.g., early 10% of states. This threshold should be chosen deliberately, and the states with an index smaller than this threshold are categorized as "early states".

- Similarly, a clear threshold is necessary for defining the "high confidence" or "high consistency", e.g., over 80% confidence and 60% consistency. With the metrics defined in Section 2.3, here, we should examine (1) the confidence of the early states in the trajectory and (2) the consistency across all states of the trajectory. Here, only the trajectory that exceeds the confidence threshold as well as the consistency threshold can be classified as a post-hoc trajectory.

In conclusion, our framework shows promise for identifying post-hoc trajectories. Meanwhile, we should note that it still needs (1) to choose particular thresholds for the precise definition of post-hoc trajectory and (2) to collect a set of reliable data to verify the effectiveness in identifying post-hoc trajectory. These are quite challenging to conduct. Although it goes beyond the scope of work, we believe investigating post-hoc trajectory in reasoning is valuable and merits exploration in future work.

B.7 Limitations and Future Directions

Scope. While the *Landscape of Thoughts* offers a practical lens on model reasoning, its current instantiation is limited to multiple-choice settings. Extending LoT to open-ended reasoning—including mathematical problem solving, code generation, and planning—requires handling less structured and more entangled reasoning paths. Two complementary threads of future work are: (i) improving accessibility by producing intuitive visual and textual explanations that help non-experts inspect and trust model behavior, and (ii) developing automated, scalable detectors of reasoning failures to improve reliability across applications.

Key challenge: synthesizing options. The central obstacle is the quality of the synthesized answer options. Human-authored distractors are carefully calibrated to be plausible, exposing distinctions between (1) correct reasoning and (2) reasonable-but-wrong reasoning (e.g., overlooking information or making arithmetic slips). In contrast, LLM-generated distractors can be implausible and thus trivially eliminated when juxtaposed with the correct option, yielding visualizations that over-emphasize the correct trace and limit diagnostic value. Moreover, LLMs may reuse similar reasoning patterns, producing near-duplicate error modes across incorrect options and reducing the comprehensiveness of the analysis.

Mitigations. To address these issues, we can elicit higher-quality distractors with state-of-the-art LLMs (e.g., OpenAI o3, Gemini 2.5 Pro) and tune sampling hyperparameters (temperature, top-p) to promote diversity and explore alternative solution trajectories.

Binary reformulation. A practical alternative is to recast multiple-choice prompts as binary (yes/no) queries. For example, the question "What is the capital of France?" can be reformulated as "Is Paris the capital of France?" with options *Yes* or *No*. Under this framing, both options remain prima facie plausible: the incorrect choice admits coherent yet flawed rationales, and the variety of "No" trajectories preserves diversity without resorting to obviously implausible distractors.

Beyond multiple choice. Although open-ended tasks are beyond the present scope, LoT is, in principle, extendable. The key requirement is to construct a candidate set of answers by querying the model (a non-trivial step that is given for free in multiple-choice tasks). Treat the ground-truth answer as one option and generate additional plausible alternatives using LLMs; LoT can then analyze the induced reasoning behaviors in these open-ended scenarios.

Case: code generation. Code generation introduces additional challenges: there is typically no single ground-truth program, and evaluation proceeds via test suites. Candidate programs are diverse and do not naturally discretize into options. We propose the following procedure: (i) sample multiple candidate solutions from the model under evaluation; (ii) score each by the number of tests passed; (iii) apply a threshold to separate more-correct from less-correct solutions; (iv) embed and cluster solutions within each partition; and (v) use cluster centroids as anchors for "correct" and "incorrect" choices. Cluster quality can be assessed with the Silhouette Score and the Davies-Bouldin Index. These anchors enable a LoT-style visualization over the solution space and provide insight into reasoning behaviors.

In summary, our visualization framework is adaptable beyond multiple-choice scenarios. To our knowledge, LoT is the first landscape visualization tool aimed at analyzing LLM reasoning; it is imperfect and remains open to improvement and extension. We believe it constitutes a small but meaningful step toward understanding and improving the reasoning processes of LLMs.

B.8 A Comparison Between Lightweight Verifier and Reward-guided Algorithms

It is worth noting that our goal is not to build a sophisticated verifier, but rather to demonstrate how the feature vectors from the landscape visualization can be effectively used.

In general, reward-guided algorithms are more computationally efficient than the path landscape. Specifically, for a reasoning path with n thoughts and c answer choices, constructing the landscape requires $n \times c$ forward passes through the reasoning model. In contrast, a reward-guided approach typically makes a single call to a reward model that evaluates the entire reasoning chain at once.

Meanwhile, it's important to consider the overhead involved in training the reward models in reward-guided algorithms. Notably, for Process-Reward Models (PRMs) [33, 61], collecting high-quality training data often requires detailed, fine-grained annotations of reasoning steps, which can be costly and time-consuming. Moreover, training a reward model (often itself an LLM) incurs significant computational expense. In contrast, our lightweight verifier is much more efficient to train, as it requires no human annotations and uses easily obtainable data.

C Related Work

Reasoning with large language models. Chain-of-Thought (CoT) prompting [58, 27] has empowered LLMs to tackle multi-step reasoning problems by generating intermediate steps before producing a final answer. Building upon CoT, numerous methods have been proposed to address various challenges, including compositional generalization [70, 25], planning [63, 18], and rule learning [71] within the CoT reasoning. Beyond solving reasoning tasks, CoT has also emerged as a foundational framework for other techniques, such as fine-tuning LLMs [67], enabling LLM-based agents [64], and facilitating test-time scaling [44]. Nevertheless, most of these approaches are developed in a trial-and-error manner, largely due to the absence of proper tools for analyzing the CoT.

Understanding chain-of-thought reasoning. There are a few studies that explore what makes CoT prompting effective by perturbing its exemplars. To be specific, Madaan and Yazdanbakhsh [34] found that the text and patterns of exemplars help CoT generate sentences resembling correct answers. Besides, Wang et al. [53] highlighted the importance of maintaining the correct order of reasoning steps, while Ye et al. [66] demonstrated that using complementary exemplars can enhance reasoning performance. Furthermore, CoT can benefit from longer reasoning chains, even without new information to the prompt [23]. Another line of research investigates CoT's general behavior [47, 39, 40, 43]. For example, CoT heavily depends on the semantic structure of the problem to perform reasoning [47], struggles with planning and unification in deductive reasoning [39], has difficulty generalizing to longer reasoning paths [40], and can be easily misled by irrelevant information in the context [43]. However, these observations are derived from specific reasoning tasks and prompt settings, limiting their applicability to other scenarios. In contrast, we introduce a general-purpose tool that allows users to analyze reasoning in their contexts.

Tools for analyzing chain-of-thought. To the best of our knowledge, the only existing tool for analyzing CoT is gradient-based feature attribution [59], which computes a saliency score for each input token based on the model's output. However, these token-level saliency scores do not directly capture the thought-level, multi-step reasoning process of LLMs. Consequently, the main finding in [59] is that CoT stabilizes saliency scores on semantically relevant tokens compared to direct prompting. Metrics designed to quantify CoT performance [8, 51] can also be used to analyze the reasoning behaviors of LLMs. For instance, Ton et al. [51] employs information gain to identify failure modes in reasoning paths, aligning with Observation 3.7 in this paper. However, our 2-D visualization offers significantly deeper insights than a single information gain metric. Additionally, the verifier derived from our tool is conceptually related to outcome-supervised reward models [11].

Measuring uncertainty and consistency in LLM reasoning. Several works in this research line compute metrics (such as confidence and perplexity) by leveraging the features from LLMs to measure and detect hallucination in reasoning [29, 9, 62]. Specifically, low confidence and high perplexity

often indicate unreliable reasoning, enabling the development of lightweight detectors to guide reasoning and mitigate hallucinations. However, these metrics have limitations [60, 69]: they can exhibit over-confidence or low perplexity in incorrect responses, their reliability relies heavily on the models' capability, and they cannot provide more comprehensive insights into the multiple reasoning trajectories. By contrast, our landscape of thoughts offers a holistic approach, integrating several existing metrics. This framework enables global qualitative analysis, including measures of perplexity, consistency, and uncertainty. In addition, the landscape of thoughts enables the development of advanced tools to enhance reasoning by using the features and metrics, as mentioned in Sec. 3.3.

D Experiment Settings

D.1 Setup

Visualizing the landscape of thoughts fundamentally relies on the decoding probability of LLMs. To this end, we adopted four open-source models with varying parameter sizes, namely `Llama-3.2-1B`, `Llama-3.2-3B`, `Llama-3.1-8B`, and `Llama-3.1-70B`. We repeatedly sample 10 times from the target LLM using the same reasoning strategy as self-consistency [55].

For visualization purposes, we randomly sample 50 questions from the testing split of each dataset and generate reasoning paths with the setup described above. For simplicity, we compute distances only between each state and all candidate answers. To visualize multiple problems in a shared space, we always place the distance to the correct answer as the first element of each feature vector. This alignment allows joint analysis across problems, as introduced in the paragraph below Equation 4. We then aggregate feature vectors from all problems into a feature matrix (Equation 2), which is passed to t-SNE to compute the pairwise distance between any two states and then outputs the 2D coordinate of each state.

For training the lightweight verifier, we randomly sample 20 questions from the training split of each dataset to obtain the feature matrix $\boldsymbol{S}$. We extract these features using three model scales: `Llama-3.2-3B`, `Llama-3.1-8B`, and `Llama-3.1-70B`. Despite the relatively small training set, it proves sufficient for our lightweight verifier, which we subsequently evaluate on the data for visualization in Sec. 3.

D.2 Datasets

AQuA [31]. This dataset develops to challenge language models' quantitative reasoning capabilities. The AQuA presents complex algebraic word problems in a multiple-choice format, where only one is correct. Each problem requires numerical computation, deep linguistic understanding, and logical inference. It provides a nuanced assessment of a model's ability to translate textual information into algebraic reasoning.

MMLU [19]. Spanning 57 distinct academic and professional domains, MMLU provides a rigorous test of language models' capabilities across humanities, social sciences, hard sciences, and technical disciplines.

StrategyQA [16]. This dataset is designed to evaluate implicit reasoning and multi-hop question answering. The dataset is characterized by yes/no questions that demand implicit reasoning strategies. Unlike straightforward factual queries, these questions require models to construct elaborate reasoning paths, showing hidden logical connections.

CommonsenseQA [46]. This dataset assesses commonsense reasoning through multi-choice questions derived from the ConceptNet knowledge graph [45]. The dataset aims to test a model's understanding of commonsense concepts and ability to make logical inferences. However, the questions often require the model to incorporate external knowledge to select the correct answer from plausible distractors.

Note that AQuA, MMLU, and StrategyQA all demand exploratory traversal of intermediate reasoning states, resulting in diverse but structured landscapes. CommonsenseQA, conversely, represents a distinct domain where answers depend on static knowledge rather than emergent reasoning pathways.

Table 1: Statistical verification of the observations in Sec. 3.

(a) Verifying Observation 3.6

	Correct	Incorrect
CoT	1.026	0.975
L2M	1.026	0.989
ToT	1.004	0.987
MCTS	1.002	0.985

(b) Verifying Observation 3.7 and 3.1

	Speed	Accuracy
CoT	0.322	84.4%
L2M	0.224	82.2%
ToT	0.205	81.6%
MCTS	0.198	75.8%

(c) Verifying Observation 3.4

	AQuA	MMLU	StrategyQA	Common SenseQA
AQuA	1.0	0.914	0.895	0.859
MMLU	0.914	1.0	0.870	0.843
StrategyQA	0.895	0.870	1.0	0.889
Common SenseQA	0.859	0.843	0.889	1.0

D.3 Decoding Algorithms

Chain of Thought (CoT) [58]. CoT elicits the LLM's reasoning capabilities by incorporating few-shot examples that demonstrate explicit reasoning steps. It provides the model with exemplar reasoning traces to guide its problem-solving process.

Zero-shot CoT [27]. The core idea of this prompt strategy lies in adding simple instructions, e.g., "Let's think step by step." to the prompt, enabling models to generate reasoning traces without assigned task-specific examples.

Least-to-Most (LtM) [70]. LtM is an innovative reasoning approach that systematically breaks down complex problems into progressively simpler subproblems. This approach mirrors human cognitive problem-solving strategies, where individuals naturally break down complex tasks into smaller, more comprehensible parts.

Tree-of-Thought (ToT) [63]. ToT expanded this concept by creating a more sophisticated, multi-branching reasoning framework. While CoT follows a linear path of reasoning, ToT introduces a more dynamic exploration, allowing models to generate multiple reasoning paths simultaneously, evaluate them, and strategically prune less promising trajectories.

Monte Carlo tree search (MCTS) [68]. MCTS is a powerful computational algorithm originally developed for game-playing strategies, particularly in complex decision-making environments like chess and Go. The method uses probabilistic sampling and tree exploration to systematically navigate potential solution spaces, balancing exploring new possibilities with exploiting promising paths. We adopt the task-agnostic node expansion and evaluation prompt from ReST-MCTS [68] to conduct our experiment across different tasks.

E Supplementary Results and Analysis

E.1 Statistical Verification of the Observations

In this part, we conduct extra experiments and statistically verify Observations 3.1, 3.4, 3.6, and 3.7, while the other Observations 3.2, 3.5, and 3.8 have been quantitatively verified by the metrics in Sec. 2.3.

To verify Observations 3.6, we calculate the convergence coefficient (e^{β}) by fitting a log-linear regression model to the sequence of distances d_i between each state and the final answer as $\log(d_i) \approx \alpha + \beta i$, where α is the intercept term; β is the slope coefficient that quantifies convergence behavior; i represents the position index in the reasoning chain. Lower values of e^{β} indicate faster convergence. For Observations 3.1 and 3.7, we measure the speed of a reasoning path moving from start to end as speed $= \frac{\|\bar{s}_n - \bar{s}0\|}{\sum j=1^n \|\bar{s}j - \bar{s}j-1\|} \in [0, 1]$, where $\bar{s}_i$ represents the 2D coordinate of the state i. Whereas Observation 3.4, we compute pairwise histogram intersection scores of the density distributions. Lower scores indicate greater dissimilarity between landscapes.

Notably, for Tab. 1(a), we found that correct paths consistently show slight divergence, while incorrect paths show more convergence (p-value = 0.008), thus verifying Obs. 3.6. As shown in Tab. 1(b), speed and accuracy correlate strongly (p-value = 9.421e-11), thus verifying Observation 3.7. This is also applicable for verifying Observation 3.1. Tab. 1(c) shows that lower scores indicate greater dissimilarity between landscapes, which verifies Observation 3.4, i.e., AQuA, MMLU, and StrategyQA are more similar, while CommonSenseQA exhibits distinct patterns.

E.2 Analysis of Reasoning Trajectory Convergence

We aim to investigate Observation 3.7 quantitatively to show its consistency with the statistical result. Specifically, we analyzed all questions from the AQuA dataset using the Llama-3.1-8B-Instruct model with the CoT method. Among the 500 reasoning trajectories (50 questions, with 10 trajectories per problem), we observed that cases where a reasoning chain initially approached the correct answer's path but later diverged to an incorrect conclusion were quite rare: only 4 questions (8% of all questions) exhibited this phenomenon, accounting for just 9 reasoning trajectories (1.8% of all trajectories). This indicates that such failure cases are infrequent within the overall set of generated reasoning trajectories.

Thoughts that diverge from the correct answer exhibit remarkable proximity at certain states. We quantify the proximity by measuring the distance between states: a smaller distance indicates that the state is closer to the correct answer. Notably, in the following example, the chain's reasoning reached a minimum distance of just 0.077 from the correct path before drifting to a final distance of 0.182. This reveals that even incorrect answers may closely track the correct reasoning at key moments.

We provide a concrete example of such a failure in the following reasoning chain for a question below, where the correct answer is B) 28%.

An example of reasoning that is close to the correct answer at intermediate thoughts but finally converges to the incorrect answer

> Q: The original retail price of an appliance was 60 percent more than its wholesale cost. If the appliance was actually sold for 20 percent less than the original retail price, then it was sold for what percent more than its wholesale cost?
> Options: A) 20% B) 28% C) 36% D) 40% E) 42%
The model's output is:
> 1. "To find the percent more than the wholesale cost for which the appliance was sold, we need to break down the calculations step by step."
> ...
> 11. "So, (0.28x / x) * 100 = 28%." → The state that is closest to the correct answer, with a distance of 0.077.
> 12. "Therefore, the appliance was sold for 28% more than its wholesale cost."
> 13. "The answer is C." → Eventually, this state reaches the incorrect answer, with distance as 0.182

E.3 Further Investigation on the Consistency Metric

In the Tab. 2, we analyze the model responses for drawing Fig. 5 and report (1) the average number of thoughts, (2) the average number of tokens in a thought, and (3) the average consistency of different thoughts.

Table 2: The relation of consistency with the number of thoughts and tokens

Model	Avg. Thoughts	Avg. Tokens	Avg. Consistency
Llama-3.2-1B	8.07	346.81	0.51
Llama-3.2-3B	11.73	439.37	0.40
Llama-3.1-8B	21.38	715.56	0.48
Llama-3.1-70B	13.55	442.72	0.51

As can be seen, the 8B/70B models produce more thoughts than the 1B/3B models; meanwhile, their intermediate states of correct chains in blue are more consistent than those of the 1B/3B model. The Pearson correlation coefficient between CoT length (thoughts) and consistency is only -0.0185, indicating a very weak negative correlation that is not approaching either +1 or -1. **Hence, higher consistency doesn't correlate with shorter chains. Fewer CoT steps do not necessarily indicate higher consistency.**

As we introduced in Sec. 2.3, the consistency metric is used to understand whether the LLM knows the answer before generating all thoughts. Here, the observation "larger models have higher consistency"

Table 3: Consistency Metrics Across Random Thoughts

Consistency	The number of random thoughts				
	2	4	8	16	32
Correct Paths	0.77	0.80	0.80	0.75	0.66
Incorrect Paths	0.90	0.92	0.92	0.79	0.79

Table 4: Accuracy and consistency on MMLU and MMLU-Pro across different models.

Model	MMLU Accuracy	MMLU Consistency	MMLU-Pro Accuracy	MMLU-Pro Consistency
Llama-3.2-1B Instruct	0.20	0.40	0.05	0.17
Llama-3.2-3B Instruct	0.46	0.41	0.30	0.26
Llama-3.1-8B Instruct	0.66	0.41	0.30	0.20
Llama-3.1-70B Instruct	**0.86**	**0.55**	**0.40**	**0.52**

actually indicates that a larger model has a higher probability of knowing its final answer in its middle steps of reasoning. We believe that this observation is new and insightful to the community.

In addition, we investigate whether the consistency is meaningful for the reasoning outcome or if it consistently decreases as the thoughts increases. We ask the Llama 3.1 8B Instruct model to generate some random thoughts, using a temperature of 0.7 to encourage more varied responses. For each of the 10 questions we select from AQuA, we then randomly combine different numbers of these thoughts to create 50 chains for each question, with the number of thoughts ranging from 2, 4, 8, 16, or 32. After generating these chains, we calculate the distance matrix and report the consistency, as shown in Tab. 3. **Notably, as the length of the chain of random thoughts increases, the consistency consistently decreases, regardless of the correctness, which justifies that consistency will not increase as n increases.**

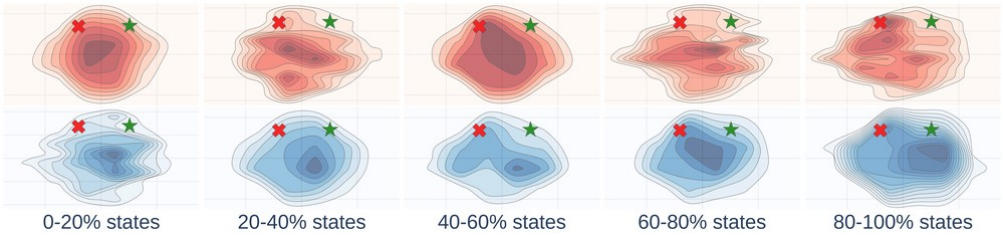
Besides, we conduct extra experiments on a harder task across model scales and show that larger models achieve higher consistency than smaller models on both easy and hard tasks. Specifically, we apply the MMLU-Pro [57] as a harder benchmark. MMLU-Pro is a more challenging version of MMLU (adopted in this work), extending the MMLU dataset by integrating more reasoning-focused questions. We sample problems from the MMLU-Pro Math subset and evaluate models of different scales, following the consistency calculation described in equation 5. The experiment results are shown as follows:

The above results show that larger models have substantially higher consistency on both the easy task (MMLU) and the hard task (MMLU-Pro) than smaller models. Here are some detailed observations: (1) Notably, on the hard task, the 70B model still has a higher consistency than the 1B/3B/8B model on either the hard task or the easy task. (2) Besides, the 70B model achieves a similar consistency on easy and hard tasks (0.55 and 0.52, respectively). (3) However, the 8B model drops significantly from easy to hard tasks (from 0.41 to 0.20).

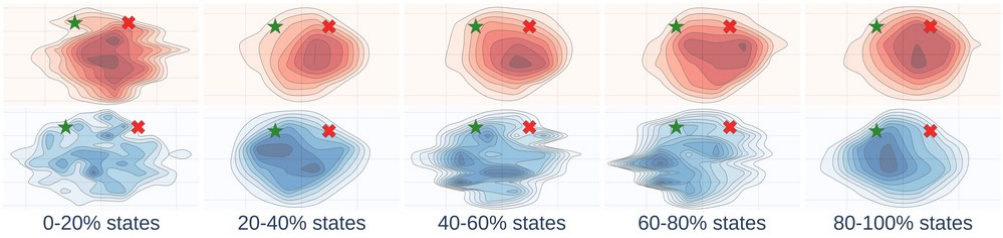
E.4 Further Discussion on the StrategyQA

The abnormal reasoning behavior, where states cluster on anchors that differ from their final answer in Fig. 4c, is not due to our visualization method but to the unstable reasoning process in the Llama-3.1-70B using CoT on StrategyQA. This model struggles to reliably represent its self-generated intermediate thoughts, presenting consistency between intermediate thoughts and final predictions, thus leading to the abnormal patterns observed.

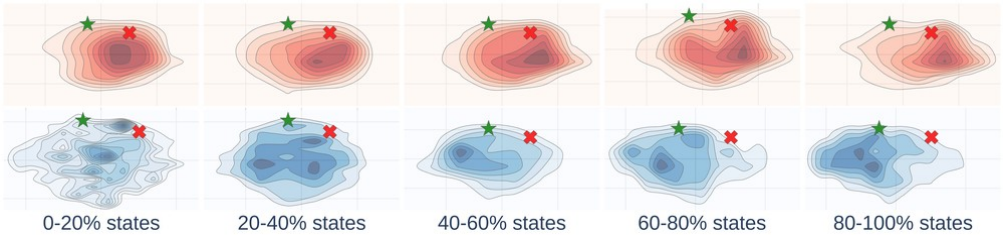
Specifically, the consistency of incorrect paths declines steadily. This highlights the model's unstable reasoning, as it fails to maintain coherent reasoning even when approaching the final answer. In addition, the landscape exhibits the highest perplexity compared to other models, indicating low confidence in its generated thoughts, which undermines the reliability of the estimated feature matrix used in our visualization.



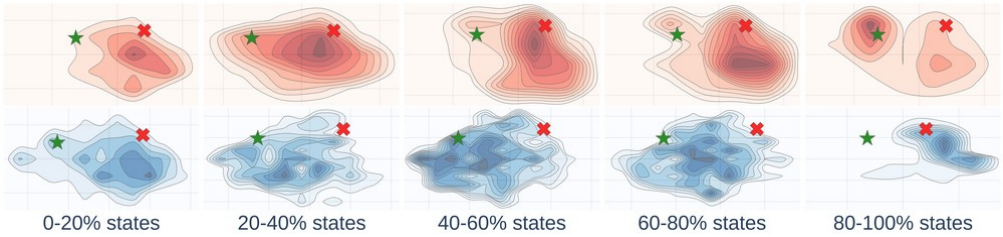
(a) Llama-3.2-1B with CoT on StrategyQA



(b) Llama-3.2-3B with CoT on StrategyQA



(c) Llama-3.1-8B with CoT on StrategyQA



(d) Llama-3.1-70B with CoT on StrategyQA

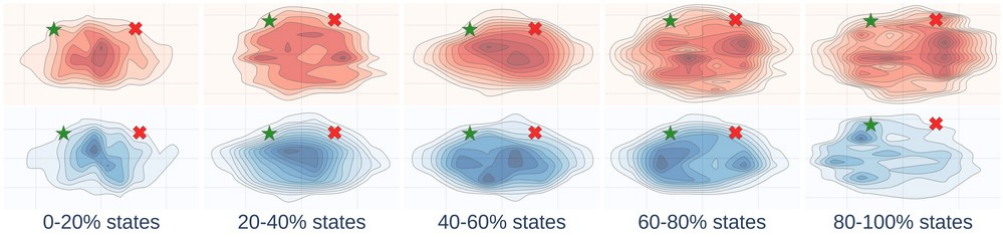
Figure 10: The landscapes of the model across scales (using CoT on the StrategyQA dataset).

Further, we provide landscape visualizations for the same dataset using other models and methods in Fig. 10 to Fig. 12. These landscapes do not exhibit the same abnormal density patterns, reinforcing that the issue is specific to Llama-3.1-70B's reasoning instability rather than a flaw in our visualization.

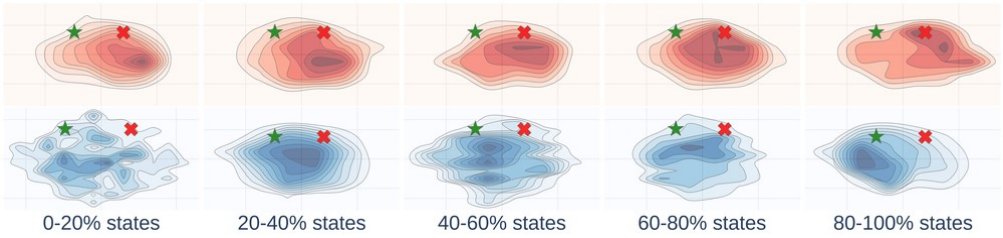
E.5 Comparing the Perplexity among Different Models

We conduct experiments to calculate the average perplexity of models in our visualization. Consistent with the prior works, we find that different models present similar perplexity when decoding the same set of CoTs. Here, we first generate a set of CoTs from the AQuA dataset using Llama-3.1-70B Instruct. Then, we use models from the same family (*i.e.*, Llama-3.2-1B Instruct, Llama-3.2-3B Instruct, Llama-3.1-8B Instruct, and Llama-3.1-70B Instruct) to compute the average perplexity on decoding the same set of CoTs. This control experiment isolates the effect of a model's inherent perplexity calculation from the variation of its generated thoughts.

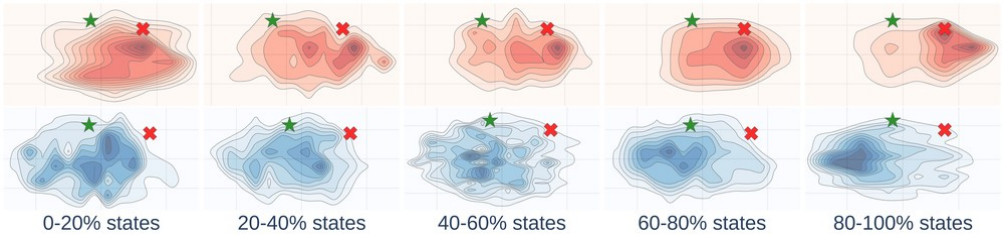
As shown in Tab. 5, while there is a slight variation in perplexity, the values all fall within a comparably narrow range (from 1.4 to 2.0). This demonstrates that for decoding the same CoTs, different models in the Llama-3 family produce similar and comparable perplexity scores.



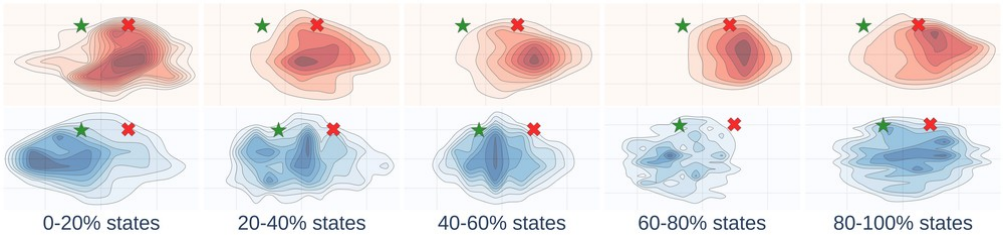
(a) Llama-3.2-1B with MCTS on StrategyQA



(b) Llama-3.2-3B with MCTS on StrategyQA



(c) Llama-3.1-8B with MCTS on StrategyQA



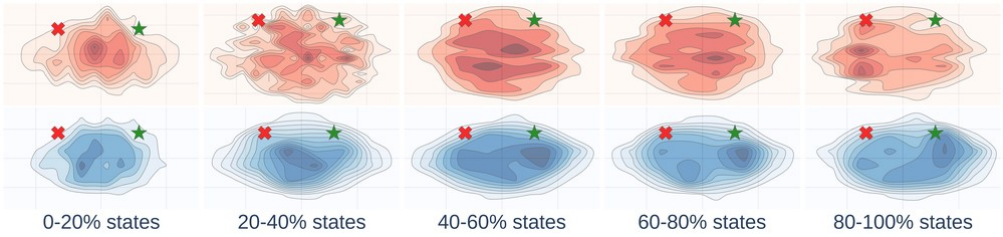
(d) Llama-3.1-70B with MCTS on StrategyQA

Figure 11: The landscapes of the model across scales (using MCTS on the StrategyQA dataset).

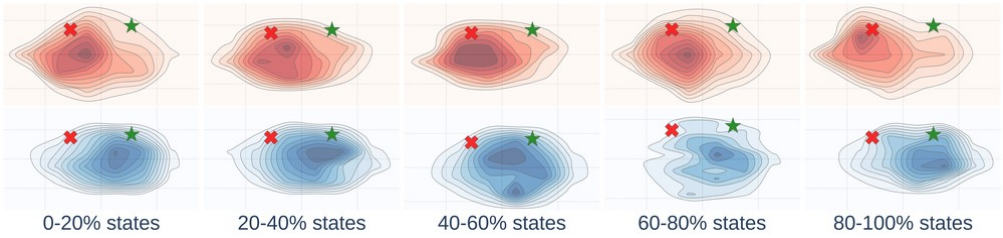
Table 5: Comparison of the perplexity of CoTs of correct and incorrect reasoning.

Model	**Avg. Perplexity (Correct CoTs)**	**Avg. Perplexity (Wrong CoTs)**
Llama-3.2-1B Instruct	1.68	1.96
Llama-3.2-3B Instruct	1.72	1.69
Llama-3.1-8B Instruct	1.61	1.49
Llama-3.1-70B Instruct	1.56	1.42

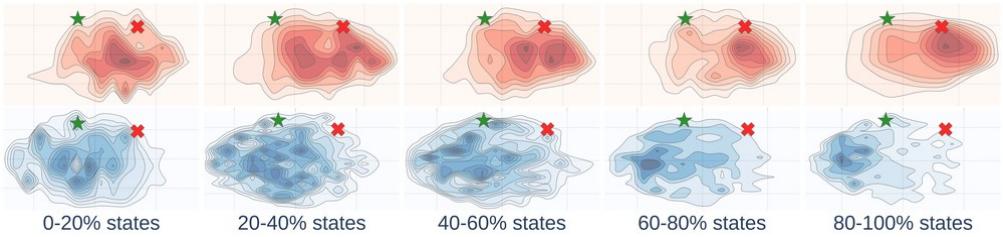
In addition, in Fig. 2, we measure the perplexity of decoding CoTs generated by the models themselves. In this context, perplexity reflects both a model's reasoning capabilities and the comprehension of its generated content. To some extent, the above findings support the validity of the comparison of perplexity across models in our study.



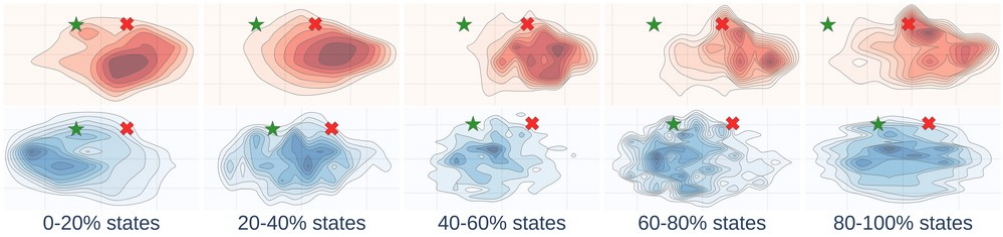
(a) Llama-3.2-1B with ToT on StrategyQA



(b) Llama-3.2-3B with ToT on StrategyQA



(c) Llama-3.1-8B with ToT on StrategyQA



(d) Llama-3.1-70B with ToT on StrategyQA

Figure 12: The landscapes of the model across scales (using ToT on the StrategyQA dataset).

Table 6: Absolute accuracy with the verifier, compared to performance in Fig. 7 (without the verifier).

(a) Across datasets

	AQuA	MMLU	StrategyQA	Common SenseQA
AQuA	63.0 (+0.7)	62.3 (+0.0)	62.3 (+0.0)	64.0 (+1.7)
MMLU	53.0 (+0.0)	53.0 (+0.0)	53.0 (+0.0)	53.0 (+0.0)
StrategyQA	41.5 (+4.5)	40.5 (+3.5)	43.0 (+6.0)	37.0 (+0.0)
Common SenseQA	54.0 (+1.0)	53.0 (+0.0)	53.0 (+0.0)	54.0 (+1.0)

(b) Across models

	1B	3B	8B	70B
1B	26.0 (+0.5)	27.5 (+2.0)	27.5 (+2.0)	27.5 (+2.0)
3B	45.5 (+0.0)	48.0 (+2.5)	51.0 (+5.5)	51.0 (+5.5)
8B	60.0 (+0.0)	60.0 (+0.0)	60.0 (+0.0)	60.0 (+0.0)
70B	74.0 (+2.0)	73.0 (+1.0)	72.5 (+0.5)	72.5 (+0.5)

E.6 Additional Experiments on the Verifier

Absolute Performance of the Verifier. In this part, we provide the absolute performance of the experiment conducted in Fig. 9. Shown as Tab. 6, the results demonstrate that our approach consistently provides improvements across different domains and models.

Variants of Verifier. In this part, we extend it into a process verifier and validate its effectiveness through additional experiments. Our lightweight verifier functions as an outcome reward

Table 7: Performance comparison of reasoning methods across model scales on the AQuA dataset, with and without verifiers.

Model	Method	Without Verifier	With Outcome Verifier	With Process Verifier
Llama-3.2-1B	CoT	0.26	0.28	0.26
	L2M	0.22	0.24	0.29
	ToT	0.35	0.38	0.35
	MCTS	0.29	0.32	0.31
Llama-3.2-3B	CoT	0.46	0.51	0.46
	L2M	0.29	0.31	0.31
	ToT	0.33	0.35	0.33
	MCTS	0.35	0.36	0.35
Llama-3.1-8B	CoT	0.60	0.63	0.60
	L2M	0.58	0.62	0.58
	ToT	0.50	0.53	0.50
	MCTS	0.50	0.51	0.50
Llama-3.1-70B	CoT	0.72	0.73	0.73
	L2M	0.72	0.72	0.73
	ToT	0.74	0.74	0.74
	MCTS	0.72	0.73	0.72

model (ORM), assessing the correctness of an entire reasoning path. Specifically, the process verifier predicts the accuracy of each reasoning state using features from the current and all previous thoughts. State accuracy reflects whether the current state is closer to the correct answer (measured by perplexity) than other answers. We then aggregate these predictions across the chain to estimate overall accuracy.

Empirically, we collect the state-wise data by comparing the state features and the correct answers, and train the process verifier. Note, we do not need to manually annotate the step-wise rewards to train conventional PRMs. Results in Tab. 7 show that this process verifier is comparable to the outcome verifier.

Comparing the lightweight verifier with existing verifiers. In the following, we compare our lightweight verifier with the other two types of existing verifiers: the LM-based verifier and the model-self verifier.

The LM-based verifier leverages another powerful LLM (not the model to do reasoning) to semantically analyze reasoning trajectories, mimicking human expert evaluation to detect errors in the trajectories. These verifiers rely on extensive, specially curated datasets (e.g., PRM800k [30]) to train a language model for process verification. Here, collecting high-quality training data often requires detailed, fine-grained annotations of reasoning steps, which can be costly and time-consuming. Moreover, training this verifier (often itself a large language model) incurs much additional computational expense. In contrast, our lightweight verifier is much more efficient to train, as it requires no human annotations and only uses easily obtainable data that is collected from the model to do reasoning.

As for the model-self verifier [29, 60], it utilizes features derived from the model itself, such as uncertainty, perplexity, or entropy, eliminating the need for an external model and enhancing efficiency in search-based methods. While these model-self verifiers are training-free and efficient, they lack the learnability to be trained and optimized, as the model is not trained on the downstream task, and thus it can be suboptimal. In contrast, our verifier is specifically trained with the downstream task's data collected from the model, ensuring greater reliability compared to model-self verifiers.

Therefore, our landscape-based lightweight verifier offers distinct advantages in terms of efficiency and reliability over the other two types of verifiers.

Ablation study on verifier. We conduct an extra ablation study on training the verifier with either consistency or 2D information. We report the accuracy of reasoning under Least-to-Most with different model scales, averaged across different datasets.

As shown in the Tab. 8, the combination of the consistency score and 2D information delivers the best overall accuracy. This shows that our verifier could utilize the complementary aspects of both kinds of features to access the reasoning chains and thus boost reasoning accuracy.

Table 8: Ablation study on data employed for training the verifier.

	1B	3B	8B	70B
Consistency only	0.21	0.31	0.59	0.71
2D information only	0.20	0.31	0.61	0.71
Consistency + 2D information	0.24	0.31	0.62	0.72

Table 9: Performance of the verifier given different numbers of sampled paths.

Sampled Paths	Consistency	2D Information	Consistency + 2D Information
1	0.32	0.32	**0.32**
10	0.32	0.32	**0.34**
20	0.32	0.30	**0.46**
30	0.32	0.36	**0.56**
40	0.32	0.34	**0.68**
50	0.32	0.30	**0.66**

E.7 Further Experiments on the Scaling Effect

We present experiments and demonstrate that combining both information sources is the best choice, with significant gains from more sampled trajectories (i.e., test-time scaling) compared to the verifier trained with either feature, as can be seen in Tab. 9. Here, we report the accuracy using the Llama-3.2-3B Instruct model on the StrategyQA dataset as follows. As can be seen, the advantages of using both information sources increase with more sampled trajectories, especially for more than 20 sampled trajectories. In contrast, verifiers trained only on consistency or 2D information peak earlier, showing no notable performance gains beyond 10 sampled trajectories.

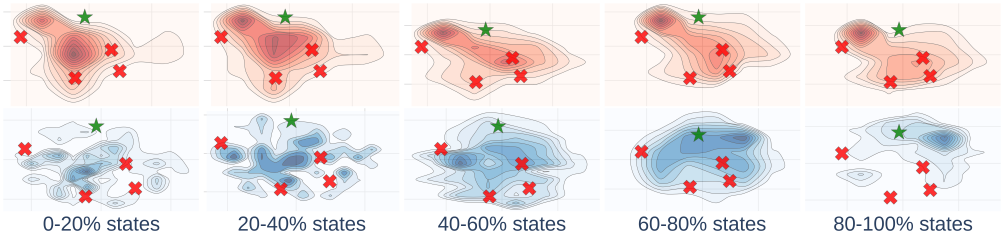
E.8 Landscapes with Different Methods of Dimensionality Reduction

t-SNE is widely adopted in non-linear projection for visualisations, which makes the plots more interpretable. Beyond t-SNE [7], several advanced dimensionality reduction techniques have been developed to improve visualization quality and efficiency. UMAP [37] outperforms t-SNE by better balancing local and global structure preservation while offering greater speed and scalability for large datasets. TriMAP [4] prioritizes both local and global preservation but tends to emphasize global structure in practice, potentially at the expense of local details. PaCMAP [56] achieves a robust balance between local and global structure preservation by incorporating neighbors, mid-near points, and further points, resulting in high-quality visualizations across diverse scenarios.

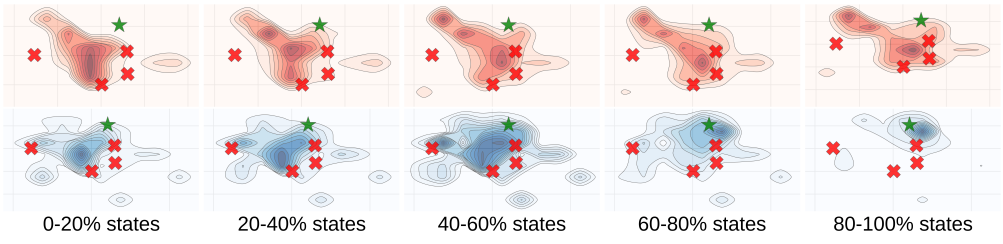
In addition, our goal is to develop a visualization tool to help users analyze the reasoning behaviors of LLMs. If necessary, we can change the adopted t-SNE to more advanced methods of dimensionality reduction. Our tool is designed to be compatible with these methods.

Next, we experiment with different dimensionality reduction methods, including t-SNE, UMAP, and PacMAP, to visualize the landscape. **Across all three visualization techniques, we consistently observe the same overarching dynamics in the reasoning process.** In the early stages (0–40% of states), the thought states are widely dispersed. As reasoning progresses, states gradually converge toward the final answer choices. Importantly, a clear distinction emerges between correct and incorrect reasoning paths, regardless of the selection of different dimensionality reduction methods. Incorrect paths tend to converge rapidly toward wrong answers early in the process, while correct paths exhibit a more gradual and deliberate progression, only clustering tightly around the correct answer in the final stages (80–100% of states).

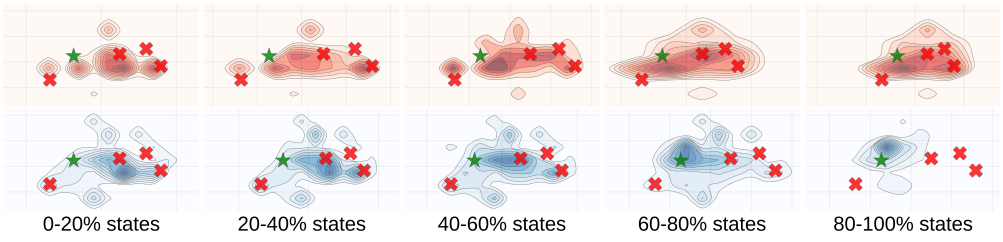
We provide landscape visualizations in Fig. 13 with different dimensionality reduction methods. While the specific geometry and density of clusters may vary between t-SNE, UMAP, and PacMAP, the fundamental narrative is unchanged: the landscape of thoughts consistently reveals that incorrect reasoning solidifies quickly, whereas correct reasoning is characterized by a slower, more refined convergence. This consistency across different dimensionality reduction algorithms demonstrates that



(a) tSNE

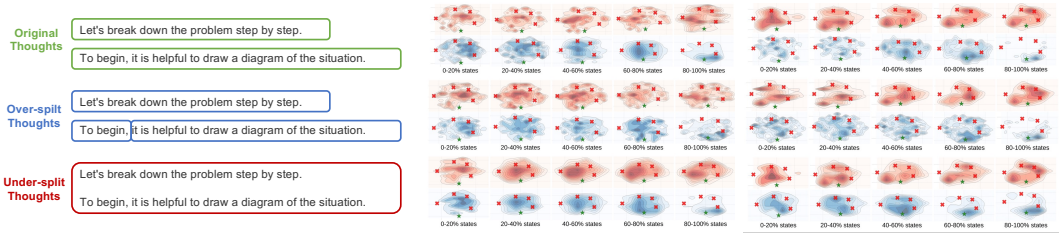


(b) PaCMAP



(c) UMAP

Figure 13: The landscapes of thought visualization with different dimensionality reduction methods (Llama-3.1-70B with CoT on AQuA).



(a) Demonstration of Sentence Tokenization (b) Llama-3.1 8B (c) Llama-3.1 70B

Figure 14: Demonstration of sentence tokenization methods for thoughts splitting.

our observations are not artifacts of a particular visualization technique, but rather reflect intrinsic properties of the model's reasoning process.

E.9 Robustness of Sentence Tokenization

To evaluate the robustness of the landscape to the split thoughts' information volume, *i.e.*, the granularity of the sentence tokenization, we conduct a controlled experiment by considering two imperfect cases in thought split, namely over-split thoughts and under-split thoughts.

Specifically, shown as Fig. 14 (a), compared to the original thoughts split that transform sentences to thoughts based on the period, over-split thoughts jointly consider the comma, resulting in additional splits. For the under-split, two adjacent thoughts are merged into one thought. We then visualize the imperfect thought splits using CoT on AQuA following the setting in Fig. 5a and Fig. 2c,

Shown in Fig. 14 (b) and (c), the landscapes are robust to the split thoughts' information volume, which are stable and consistent with our observations. Notably, for over-split thoughts, the states are more visually diverse but eventually converge to the answers. Whereas under-split thoughts, the states show a more compact pattern and exhibit a clear convergence trend toward the answer.

F Visulizations

In this part, we provide the full visualization of the verifier performance and landscapes.

In Fig. 15 to Fig. 18, we visualize the average voting accuracy (%) of different LLMs reasoning with and without verification on various datasets and methods. In Fig. 19 to Fig. 22, we display the landscape of different models on various datasets using four methods. We also provide case studies by visualizing the landscape with corresponding states in Fig 23 to Fig. 26.

In addition, we provide the landscape of thoughts on the latest reasoning model. Specifically, we conduct experiments on the DeepSeek-R1-Distill models [17] (Llama-70 B and Qwen-1.5 B). As shown in Fig. 27 and Fig. 28, the landscape of the reasoning model also aligns with the observation drawn from the general-purpose model, but exhibits more complex reasoning patterns, such as self-evaluation and back-tracking.

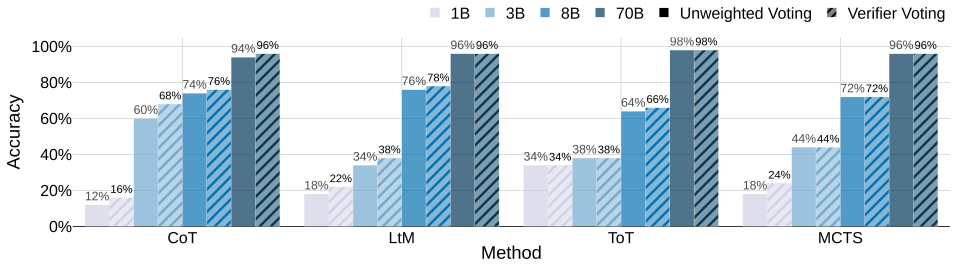


Figure 15: Average voting accuracy (%) of reasoning with and without verification on AQuA.

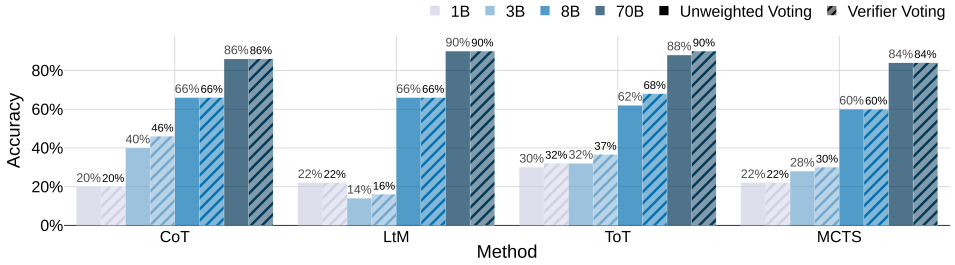


Figure 16: Average voting accuracy (%) of reasoning with and without verification on MMLU.

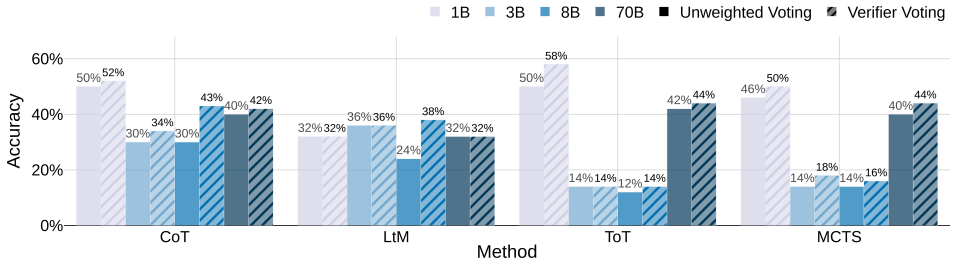


Figure 17: Average voting accuracy (%) of reasoning with and without verification on StrategyQA.

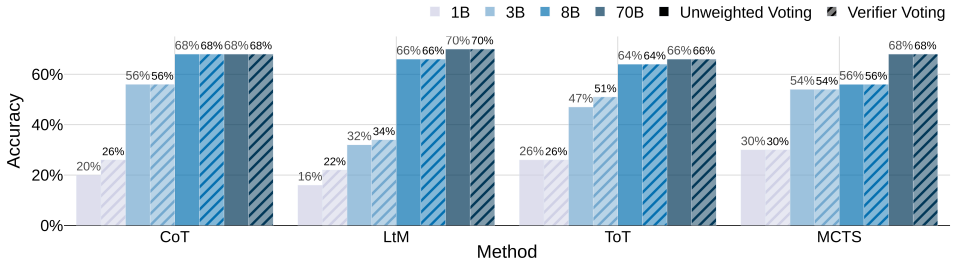
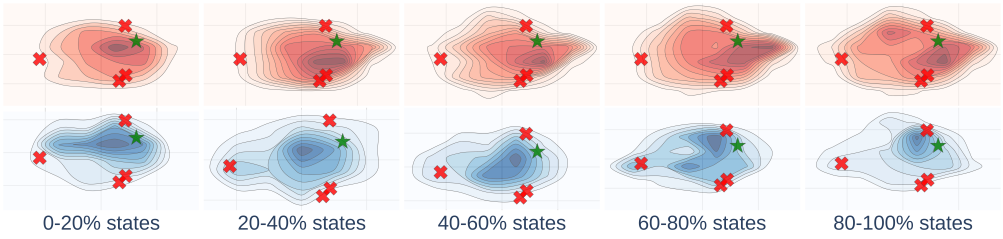
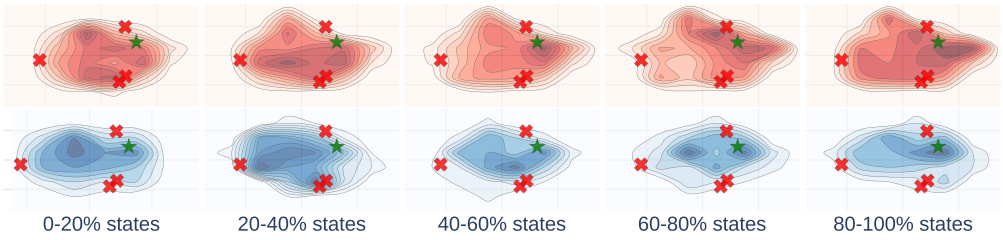


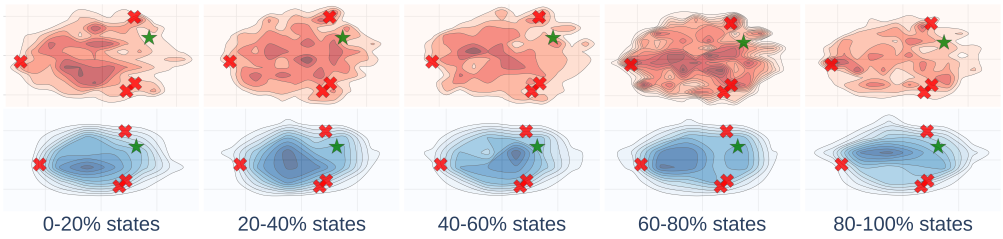
Figure 18: Average voting accuracy (%) of reasoning with and without verification on CommonSenseQA.



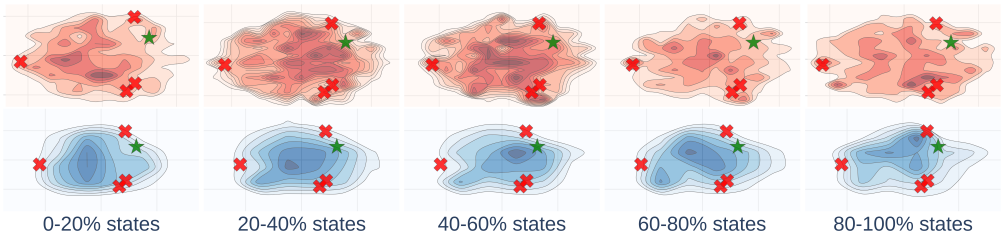
(a) Llama-3.2-1B with CoT on AQuA



(b) Llama-3.2-1B with LtM on AQuA

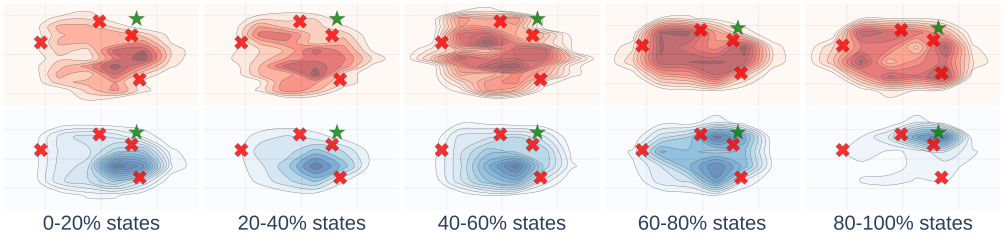


(c) Llama-3.2-1B with ToT on AQuA

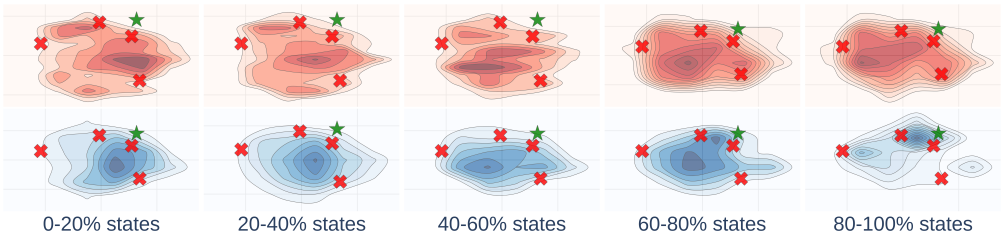


(d) Llama-3.2-1B with MCTS on AQuA

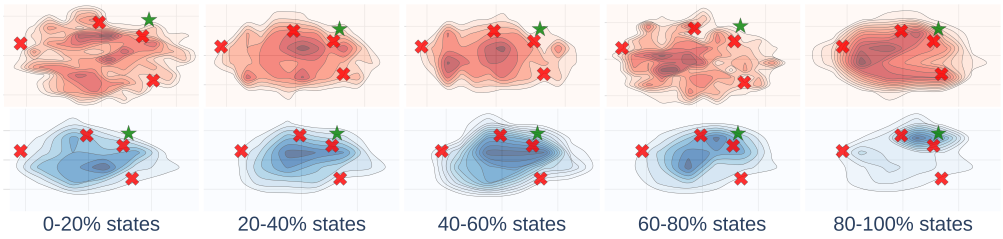
Figure 19: The landscapes of various reasoning methods (using Llama-3.2-1B on the AQuA dataset).



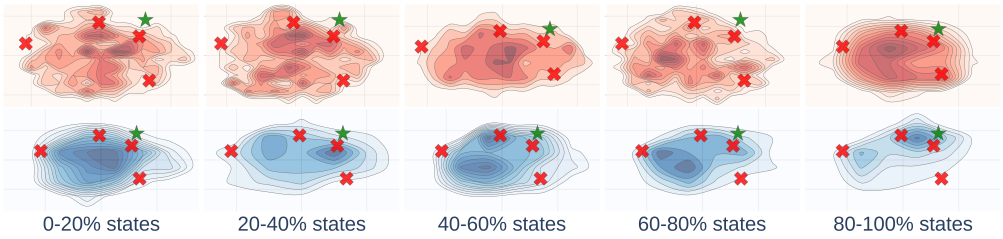
(a) Llama-3.2-3B with CoT on AQuA



(b) Llama-3.2-3B with LtM on AQuA

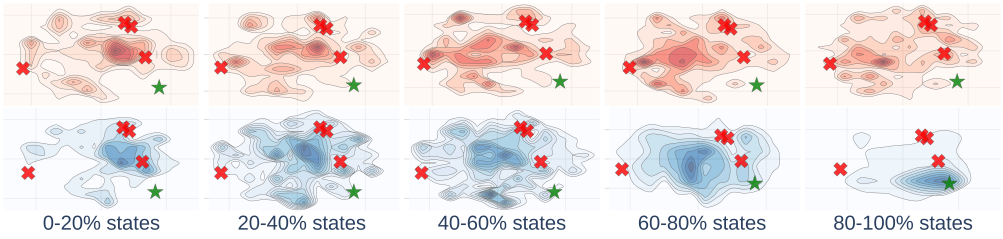


(c) Llama-3.2-3B with ToT on AQuA

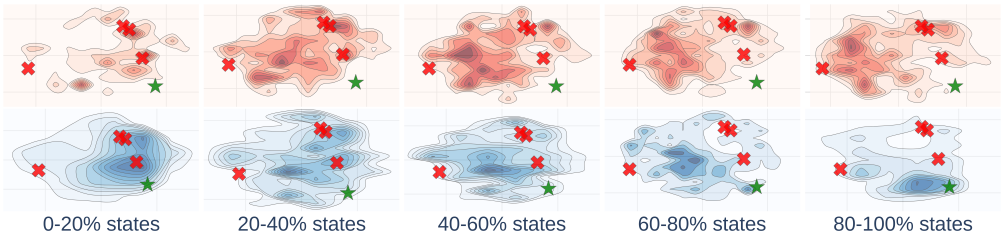


(d) Llama-3.2-3B with MCTS on AQuA

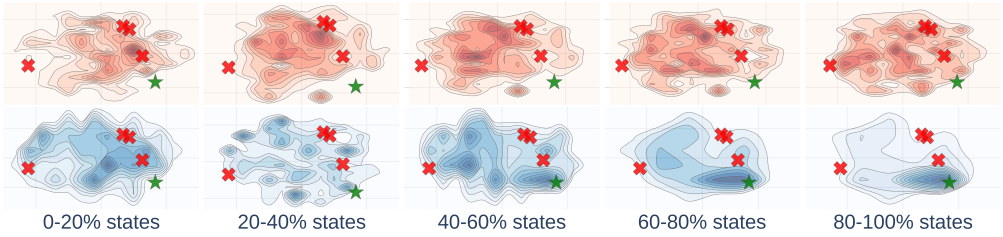
Figure 20: The landscapes of various reasoning methods (using Llama-3.2-3B on the AQuA dataset).



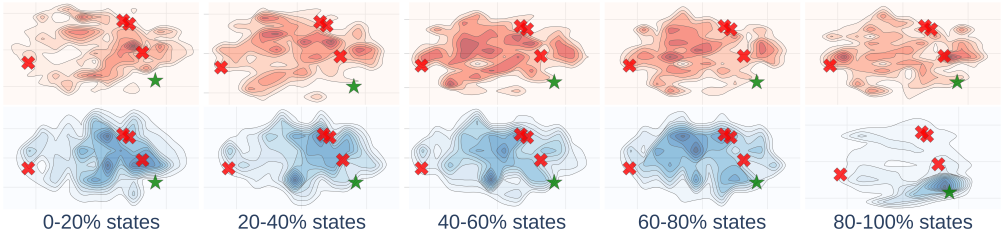
(a) Llama-3.1-8B with CoT on AQuA



(b) Llama-3.1-8B with LtM on AQuA

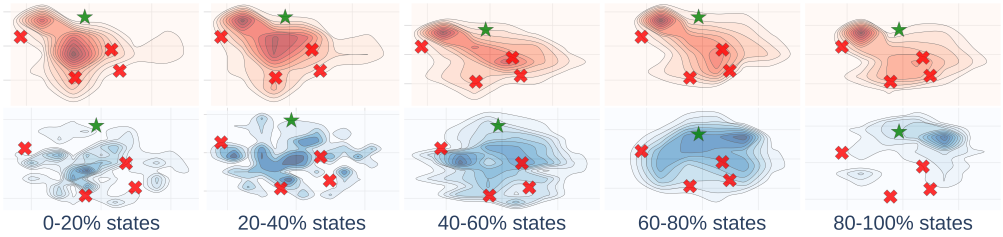


(c) Llama-3.1-8B with ToT on AQuA

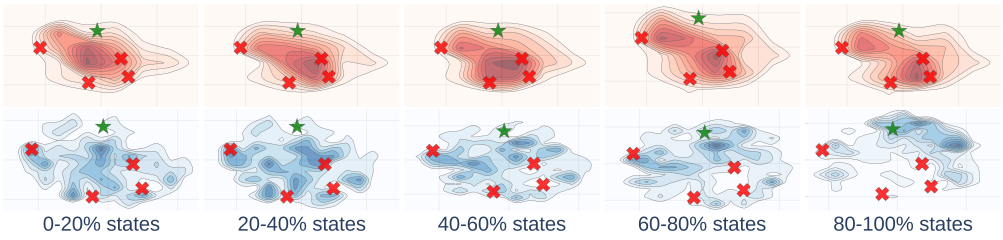


(d) Llama-3.1-8B with MCTS on AQuA

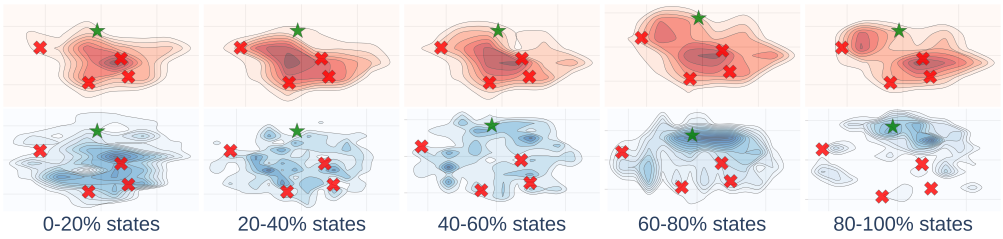
Figure 21: The landscapes of various reasoning methods (using Llama-3.1-8B on the AQuA dataset).



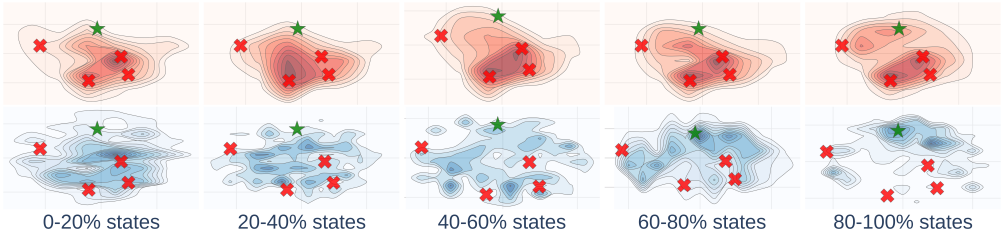
(a) Llama-3.1-70B with CoT on AQuA



(b) Llama-3.1-70B with LtM on AQuA



(c) Llama-3.1-70B with ToT on AQuA



(d) Llama-3.1-70B with MCTS on AQuA

Figure 22: The landscapes of various reasoning methods (using Llama-3.1-70B on the AQuA dataset).

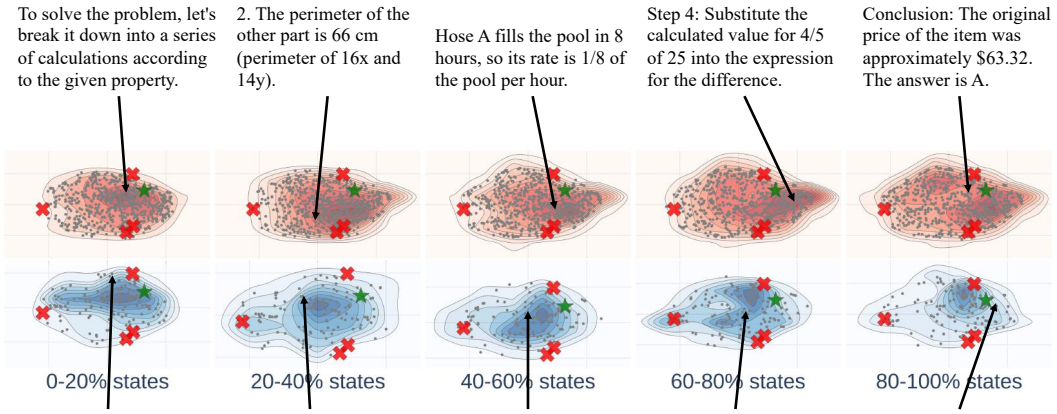


Figure 23: Case Study: Landscape of thoughts of Llama-3.2-1B on AQuA using CoT.

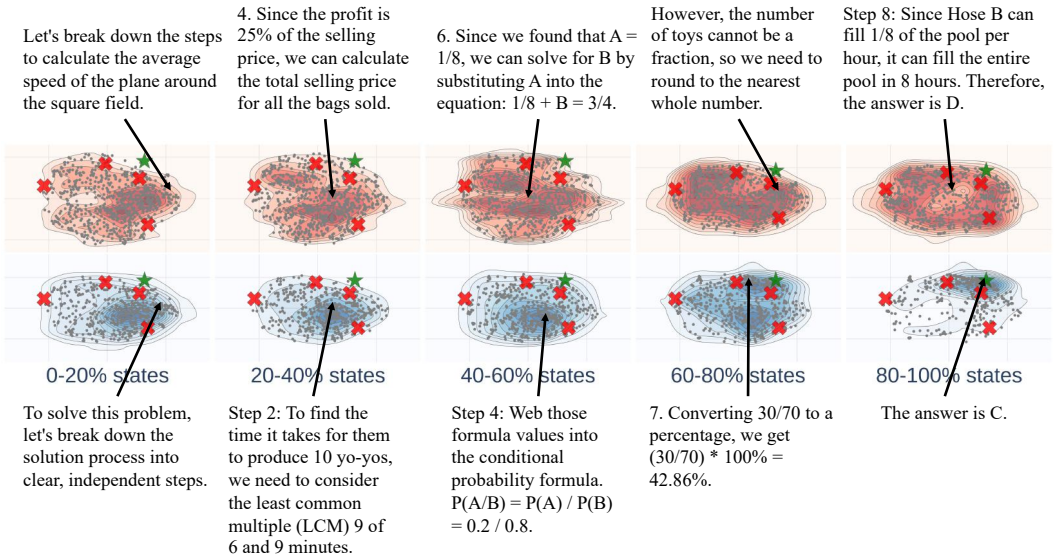


Figure 24: Case Study: Landscape of thoughts of Llama-3.2-3B on AQuA using CoT.

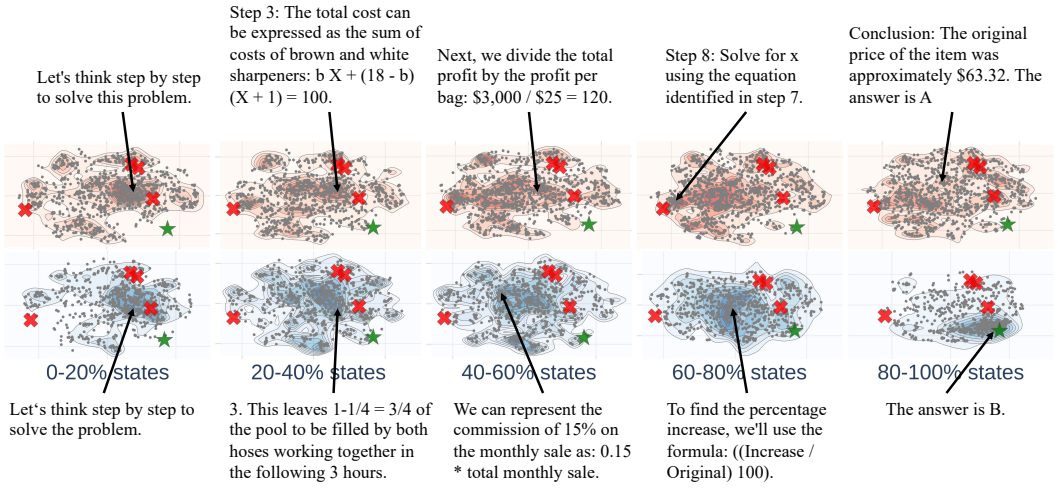


Figure 25: Case Study: Landscape of thoughts of Llama-3.1-8B on AQuA using CoT.

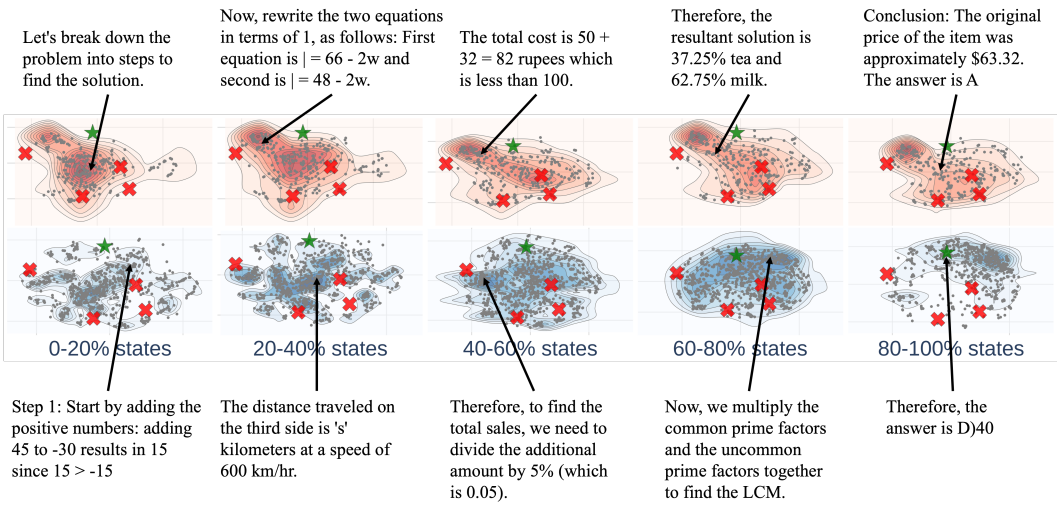


Figure 26: Case Study: Landscape of thoughts of Llama-3.1-70B on AQuA using CoT.

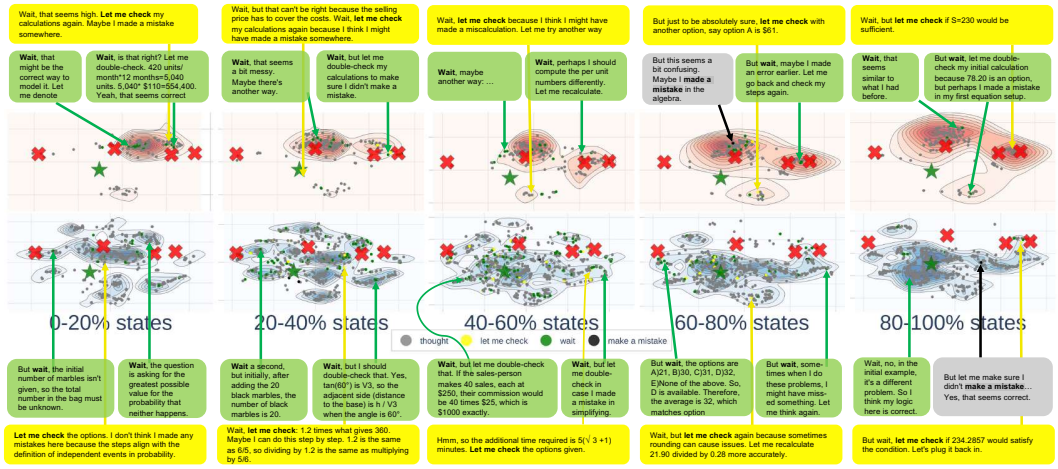


Figure 27: Landscape of DeepSeek-R1-Distill-Llama-70B using CoT on AQuA.

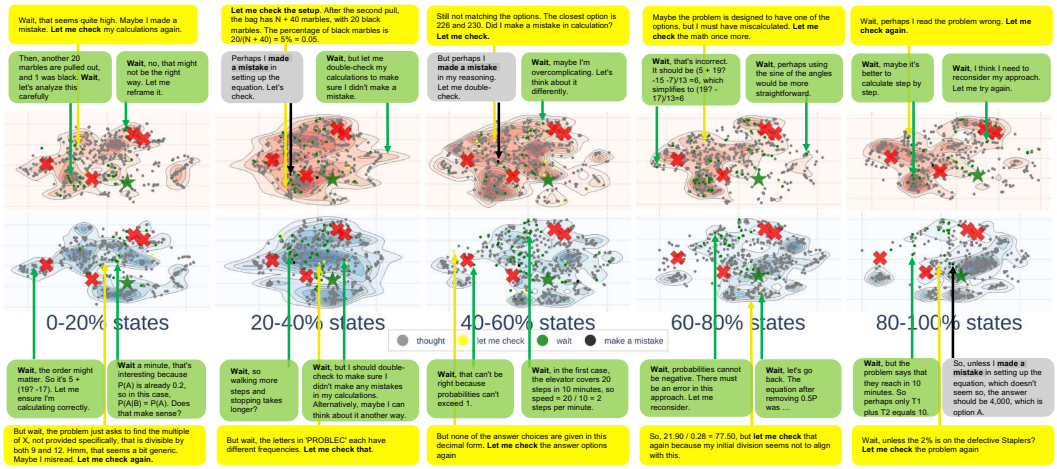


Figure 28: Landscape of DeepSeek-R1-Distill-Qwen-1.5B using CoT on AQuA.

G NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We have summarized the position and key contributions of the paper in the abstract and introduction parts.

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Sec. 5.

3. **Theory assumptions and proofs**

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

   Answer: [NA]

   Justification: We do not conduct theoretical analysis, and we explain each mathematical equation in detail.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The experiment details of implementation are introduced in Sec. 3 and Sec. 4.

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

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

   Answer: [Yes]

   Justification: The source files are publicly available at an anonymous link.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The experiment details of implementation are introduced in Sec. 3 and Sec. 4.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We conduct evaluations and repeat the reasoning 5 times for each question to obtain consistent and reliable results by averaging.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The computed requirements are introduced in Sec. 3 and Sec. 4.

9. **Code of ethics**

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

Answer: [Yes]

Justification: We have carefully checked the NeurIPS Code of Ethics and confirmed that our paper obeys it.

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The broader impacts are introduced in Appendix A.

11. **Safeguards**

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

    Answer: [NA]

    Justification: This work does not release new models.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: In the paper, we have introduced the resources of the models and datasets used.

13. **New assets**

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

    Answer: [NA]

    Justification: This work does not introduce new assets.

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

    Justification: The paper is not about crowdsourcing experiments or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [NA]

    Justification: The paper is not about research with human subjects

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this work does not involve LLMs as any important, original, or non-standard components.