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

Zhanke Zhou [1 2 *] **Zhaocheng Zhu** [3 4 *] **Xuan Li** [1 *] **Mikhail Galkin** [5] **Xiao Feng** [1]
Sanmi Koyejo [2] **Jian Tang** [3 6] **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-the first visualization tool for users to inspect the reasoning paths of chain-of-thought and its derivatives on any multi-choice dataset. Specifically, we represent the states in a reasoning path as feature vectors that quantify their distances to all answer choices. These features are then visualized in two-dimensional plots using t-SNE. Qualitative analysis shows that 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 our tool to a model that predicts any property they observe. We showcase this advantage by adapting our tool to a lightweight verifier, which significantly improves reasoning by evaluating the correctness of reasoning paths. 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 with their broad spectrum of capabilities. In particular, several useful applications of LLMs, such as tool use (Schick et al., 2023), retrieval-augmented generation (Lewis et al., 2020), and agents (Yao et al., 2023b), heavily rely on their capability of step-by-step reasoning (Wei et al., 2022; Kojima et al., 2022). Although many base models, *e.g.*, OpenAI o1 (Jaech et al., 2024), and decoding algorithms, *e.g.*, test-time scaling-up search (Snell et al., 2024), have been introduced to advance the performance of LLMs on these applications, the underlying *reasoning behavior* of LLMs remains unclear to the community. This hinders the development of algorithms and poses potential risks at deployment (Anwar et al., 2024).

A few attempts (Wang et al., 2023a; Saparov and He, 2023; Saparov et al., 2023; Dziri et al., 2024) have been made to understand the reasoning capacity of LLMs. Nevertheless, these findings are often tied to certain decoding algorithms and tasks, which may not be so instructive for users working with their own algorithms and tasks. Instead, there is a strong demand for tools that can be applied to analyze the reasoning behavior of LLMs in the users' scenarios. We foresee that such tools will benefit three groups of practitioners: 1) engineers can iterate their solutions faster based on the feedback from the tool; 2) researchers can improve decoding algorithms based on insights revealed by the tool; 3) most importantly, safety researchers can utilize the tool to monitor, understand, and improve the behavior of LLMs.

We made a small but meaningful step towards the above goal by introducing the *landscape of thoughts*, a tool for visualizing the reasoning paths produced by chain-of-thought and other step-by-step reasoning algorithms. Given any multi-choice reasoning dataset, our tool visualizes the distribution of intermediate states and any reasoning path of interest *w.r.t.* the answer choices, which enables users to uncover reasoning patterns of LLMs in both success and failure cases (Fig. 1). The core idea is to characterize the textual states in a reasoning path as features that quantify their distances to all answer choices. These distances are estimated by the perplexity metric, with the same LLM to generate thoughts and explain for itself. The state features are then projected to a two-dimensional space via t-SNE (van der Maaten and Hinton, 2008), a non-linear dimensionality reduction method that preserves manifolds in original high-dimensional space.

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

The second AI for MATH Workshop at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada.

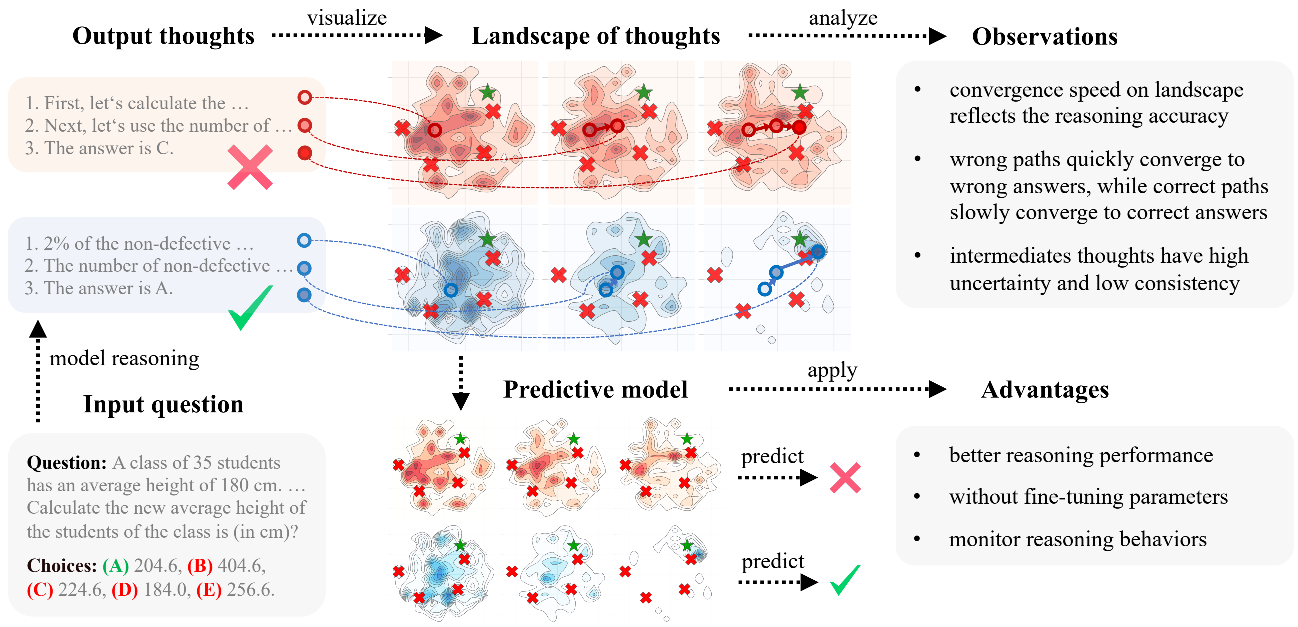

Output thoughts
visualize
Landscape of thoughts
analyze
Observations
1. First, let's calculate the …
2. Next, let's use the number of …
3. The answer is C.
1. 2% of the non-defective …
2. The number of non-defective …
3. The answer is A.
convergence speed on landscape reflects the reasoning accuracy
wrong paths quickly converge to wrong answers, while correct paths slowly converge to correct answers
intermediates thoughts have high uncertainty and low consistency
model reasoning
Input question
Predictive model
apply
Advantages
Question: A class of 35 students has an average height of 180 cm. … Calculate the new average height of the students of the class is (in cm)?
Choices: (A) 204.6, (B) 404.6, (C) 224.6, (D) 184.0, (E) 256.6.
predict
predict
better reasoning performance
without fine-tuning parameters
monitor reasoning behaviors

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 algorithm. In addition, a predictive model is applied to predict the correctness of landscapes and can help improve reasoning.

We examine our tool with different combinations of model sizes, decoding algorithms, and benchmark datasets. Our tool reveals several qualitative observations regarding the reasoning behaviors of LLMs. Some notable observations include: 1) the convergence speed of reasoning paths towards correct answers reflects the accuracy, no matter what base model, decoding algorithm, or dataset is used; 2) the convergence speed of reasoning paths in success and failure cases is distinct, indicating that we may use the convergence speed of a reasoning path 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. To our knowledge, these observations have not been reported by previous works that analyze chain-of-thought mostly based on performance metrics.

Since our tool is built on the 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 (Cobbe et al., 2021). Even though this verifier is lightweight compared to most LLM-based verifiers, it consistently improves the reasoning performance on most combinations of models, decoding algorithms, and datasets in our experiments. Hence, users can further leverage this advantage to predict other potential properties that they discover in their own scenarios.

In summary, our main contributions are three-fold:

- We introduce the first visualization tool for inspecting the reasoning dynamics of different LLMs and decoding algorithms on any multi-choice reasoning dataset (Sec. 2).
- Our tool reveals several observations regarding the reasoning behaviors of different models, algorithms, and datasets, offering new insights into the reasoning (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 parameters (Sec. 4).

2. Visualizing Multi-step Reasoning of LLMs

This section outlines a general framework for language models and reasoning algorithms compatible with our tool (Sec. 2.1), demonstrates how it visualizes reasoning by projecting thoughts into a two-dimensional space (Sec. 2.2), and introduces metrics for quantitative analysis (Sec. 2.3).

2.1. Problem Formulation

Our goal is to *visualize* the reasoning process of LLMs across a variety of problem types. To achieve this, we aim for a formulation that is sufficiently general to encompass a wide range of use cases. Specifically, we focus on datasets consisting of multiple-choice questions, where each sample $(x, y, \mathcal{C})$ 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 textual format. The proposed visualization tool applies to the following language models and reasoning algorithms.

Language models. To explore the landscape of thoughts generated by an LLM $p_{\text{LLM}}(\cdot)$, it is necessary for the model to produce diverse reasoning paths for solving a given problem. This requires the LLM to support sampling during inference $\hat{y} \sim p_{\text{LLM}}(y|x, \mathcal{C})$. For chain-of-thought reasoning, thoughts are sampled autoregressively as $\hat{t}_i \sim p_{\text{LLM}}(t_i|x, \mathcal{C}, \hat{t}_1, \ldots, \hat{t}_{i-1})$. Namely, each thought $\hat{t}_i$ is conditioned on the problem 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 reasoning paths, 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 generation $\hat{y}$. These two requirements are generally satisfied by most open-source LLMs, such as Llama (Dubey et al., 2024), Mistral (Jiang et al., 2024), and DeepSeek (Liu et al., 2024). However, proprietary LLMs, such as GPT-4 (Achiam et al., 2023) and Gemini (Team et al., 2023), are excluded as they do not support likelihood estimation.

Reasoning algorithms. While there are many approaches to solving reasoning problems with LLMs (Creswell et al., 2022; Kazemi et al., 2023), this work focuses on chain-of-thought (CoT) (Wei et al., 2022) and its derivatives (Zhou et al., 2023; Yao et al., 2023a), owing to their widespread use and development. These decoding algorithms generally guide the model in generating a structured path of intermediate reasoning thoughts before arriving at the final answer. Note that to visualize a large number of reasoning thoughts effectively, these thoughts should be automatically parsed into distinct units (*e.g.*, via sentence tokenization). This requirement is typically satisfied by most variants of CoT. We also empirically verify the robustness of our tool if this requirement does not hold (please see Appendix D.2).

2.2. Landscape of Thoughts

Given a collection of reasoning paths generated by an LLM, our tool seeks to visualize how different paths 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 2D coordinates. To address this gap, we first utilize the same LLM to represent intermediate states as numerical vectors. These state vectors are then projected into a 2D space for visualization. For simplicity, we use the notation 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 path 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 feature vectors using the likelihood function of the LLM. Specifically, the k-dim feature vector $\boldsymbol{s}_i$ for state s_i is defined as follows:

$$\boldsymbol{s}_i = [d(s_i, c_1), d(s_i, c_2), \ldots, d(s_i, c_k)]^\top, \qquad (1)$$

where $d(s_i, c_j)$ measures the *distance* between state s_i and choice c_j. In this context, the vector $\boldsymbol{s}_i$ indicates the relative distances from the state s_i to all possible choices $\{c_j\}_{j=1}^k$. To reduce the effect of length on choices, we implement the distance calculation of $d(s_i, c_j)$ through the perplexity metric (Shannon, 1948; Manning, 1999) shown as below: [1]

$$d(s_i, c_j) = p_{\text{LLM}}(c_j|s_i)^{-1/|c_j|}, \qquad (2)$$

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. We further normalize the vector $\boldsymbol{s}_i$ for a unit L1 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 $\boldsymbol{c}_j$ for a choice c_j in a manner consistent with the perplexity, namely:

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

For r paths, each with n states, we compute the feature vectors for all $r \cdot n$ states. [2] Together with the feature vectors of k choices, we obtain a feature matrix $\boldsymbol{S} \in \mathbb{R}^{k \times (r \cdot n + k)}$:

$$\boldsymbol{S} = [\boldsymbol{s}_1^{(1)}, \ldots, \boldsymbol{s}_n^{(1)}, \ldots, \boldsymbol{s}_1^{(r)}, \ldots, \boldsymbol{s}_n^{(r)}, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_k]. \qquad (4)$$

Note that a sufficiently large number of paths is necessary to generate a comprehensive visualization of the reasoning landscape. However, visualizing all samples in a dataset under this setting incurs a significant computational cost. In practice, we found it more efficient to visualize d paths with $\frac{r}{d}$ samples projected into the same space. This approach retains much of the visualization quality while substantially reducing the number of paths required for each sample. The key idea is to rearrange the order of choices such that the correct answer consistently aligns with the same dimension in the k-dimensional feature space across all the r samples.

Visualization. After constructing the feature matrix $\boldsymbol{S}$, we project the states and choices into a 2D space for visualization. This dimensionality reduction step can be accomplished using various existing algorithms (Pearson, 1901;

[1] The perplexity can also be expressed as $\text{PPL}(c_j|s_i) = \exp\left(-\frac{1}{|c_j|}\sum_{t=1}^{|c_j|}\log p_{\text{LLM}}(c_j[t]|s_i, c_j[:t])\right)$.

[2] Our tool can also be applied to paths with different numbers of states. We assume n states for demonstrations.

van der Maaten and Hinton, 2008; McInnes et al., 2018). In this study, we employ t-SNE (van der Maaten and Hinton, 2008) due to its ability to preserve the underlying manifolds of the original high-dimensional space and its robustness to a wide range of transformations. By applying t-SNE to the k-dim $\boldsymbol{S}$, we obtain the 2-dim coordinates $\bar{\boldsymbol{S}} \in \mathbb{R}^{2\times(rn+k)}$. Note that the two axes in the landscape visualization correspond to reduced dimensions from the original spaces. This original space captures the full answer space for problem-solving, with each state's coordinates reflecting its relative distance to different answers. The coordinates of the states define a discrete density function in the 2D space. To create a more intuitive and visually interpretable representation, we smooth this density function using a Parzen window estimator (Silverman, 2018). Given a coordinate $\bar{\boldsymbol{v}}$, the formulation of the smoothed density is presented as follows, where the σ controls the radius of Gaussian kernels:

$$p(\bar{\boldsymbol{v}}) = \frac{1}{rn} \sum_{\bar{\boldsymbol{s}} \in \bar{\boldsymbol{S}}} \exp\left(-\frac{||\bar{\boldsymbol{v}} - \bar{\boldsymbol{s}}||^2}{2\sigma^2}\right). \tag{5}$$

2.3. Metrics

Besides the qualitative 2D visualization, we introduce three quantitative metrics to help understand the behavior of the LLM at different reasoning steps. All these metrics are defined on the intermediate states introduced 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 $\boldsymbol{s}_i$ and $\boldsymbol{s}_n$ agree

$$\text{Consistency}(s_i) = \mathbb{1}(\arg\min \boldsymbol{s}_i = \arg\min \boldsymbol{s}_n). \tag{6}$$

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 $\boldsymbol{s}_i$ (note $\sum_{d\in\boldsymbol{s}_i} d = 1$)

$$\text{Uncertainty}(s_i) = -\sum_{d\in\boldsymbol{s}_i} d \cdot \log d. \tag{7}$$

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

$$\text{Perplexity}(t_i) = p_{\text{LLM}}(t_i|s_{i-1})^{-1/|t_i|}. \tag{8}$$

3. Results and Observations

In this section, we utilize the landscape of thoughts to analyze the reasoning behavior of LLMs. Specifically, we conduct a comprehensive evaluation and extract several observations by comparing the landscape of thoughts across three dimensions: (1) various *reasoning algorithms* in Sec. 3.1, (2) different *reasoning tasks* in Sec. 3.2, and (3) diverse scales of *language models* in Sec. 3.3.

To help understand the qualitative visualizations, we quantitatively calculate the consistency and uncertainty of states, as well as the perplexity of thoughts, all previously introduced in Sec. 2.3. Unless stated otherwise, we employ Llama-3.1-70B with CoT as the default configuration in evaluations. Note that all the visualizations are built upon the model's estimation of their intermediate thoughts.

3.1. Comparison across Reasoning Algorithms

Setup. We evaluate the default model with four reasoning algorithms: chain-of-thought (CoT) (Wei et al., 2022), least-to-most (LtM) (Zhou et al., 2023), MCTS (Zhang et al., 2024), and tree-of-thought (ToT) (Yao et al., 2023a). We run these algorithms on 50 problems randomly selected from the AQuA dataset. The corresponding landscapes are presented in Fig. 2, which yields the following observations. Further discussion, detailed experimental settings, and additional results can be found in Appendix B, C, and D, respectively.

Observation 3.1 (*The landscapes converge faster to the correct answers are of higher reasoning accuracy*)**.** By comparing the four groups of landscapes in Fig. 2, 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 path approaching one answer. As can be seen from Fig. 2, different reasoning algorithms present diverse landscapes. Generally, methods with more scattered landscapes (converge slower) present lower accuracy than those that converge faster.

Observation 3.2 (*Wrong paths quickly converge to wrong answers, while correct paths slowly step to correct answers*)**.** By comparing the landscapes of failure and success paths, it is found that the failure paths usually converge to the wrong answers at earlier states of reasoning, e.g., 20-40% states. By contrast, the states in the success paths 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 model perspective), while the correct answers are usually determined at the end of reasoning paths.

Observation 3.3 (*Compared to failure paths, the intermediate states in correct paths have higher consistency w.r.t. the final state*)**.** By comparing the consistency plots in Fig. 2, we found that the model generally has low consistency between the intermediate states and the final state. Notably, the consistency of wrong paths is significantly lower than that of correct paths. This implies that the reasoning process can be quite unstable. Even though decoding algorithms like CoT and LtM are designed to solve a problem directly (without explorations), the generated thoughts by these methods do not consistently guide the reasoning path to the answer.

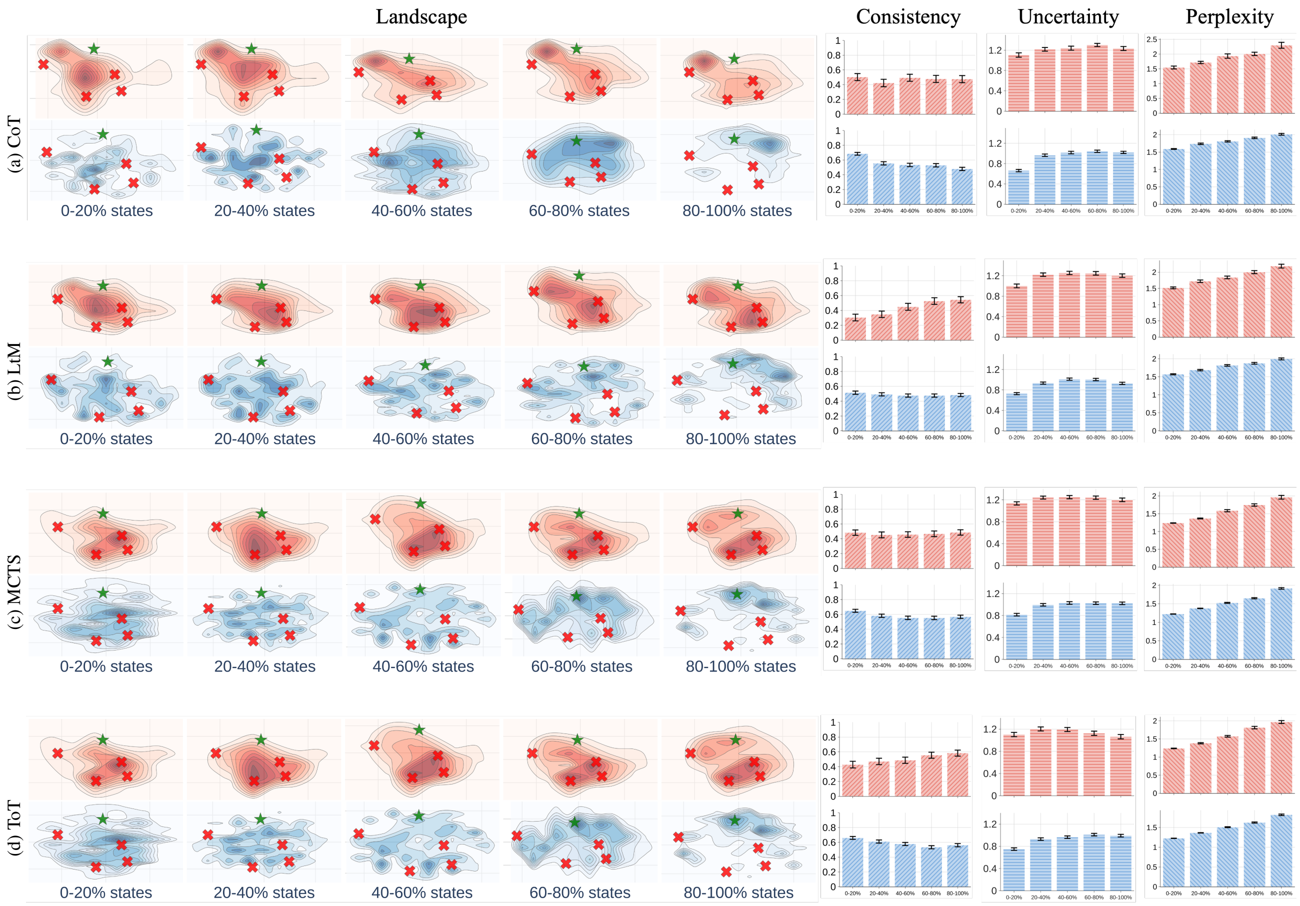


Figure 2: Comparing the landscapes and corresponding metrics of four reasoning algorithms (using Llama-3.1-70B on the AQuA dataset). Through the reasoning progression, spanning from early (0-20% states) to the later stages (80-100% states), the visualization shows correct cases (bottom row in blue) with incorrect cases (top row in red). Metrics are calculated *w.r.t.* each bin, *e.g.*, 20% - 40% of states. Note that darker regions represent a higher density of states, with ✖ indicating incorrect answers and ★ marking correct answers. The accuracy of reasoning for the four subfigures is: (a) 84.4%, (b) 82.2%, (c) 75.8%, and (d) 81.6%, respectively.

3.2. Comparison across Reasoning Tasks

Setup. Besides the AQuA, we include MMLU, CommonsenseQA, and StrategyQA datasets. We run the base model with CoT on 50 problems per dataset. The observations follow are derived from the landscapes in Fig. 3. More visualization cases can be found in Appendix E.

Observation 3.4 (*Similar reasoning tasks exhibit similar landscapes*)**.** The landscapes of AQuA, MMLU, and StrategyQA exhibit organized search behavior with higher state diversity, while CommonSenseQA presents concentrated search regions, reflecting direct knowledge retrieval 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. 3 show that path 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 paths show increasing consistency that surpasses incorrect paths at around 60% states, while incorrect paths show decreasing consistency. However, extending beyond the typical three-step requirement (Geva et al., 2021), the later stages (60-100% states) show increasing perplexity as well as lower uncertainty.

3.3. Comparison across Language Models

Setup. In this part, we study several LLMs' behavior across different parameter scales (1B, 3B, 8B, and 70B). We run each model with CoT on 50 problems from the AQuA dataset. The landscapes of these models are shown in Fig. 4. We also provide case studies on the reasoning models (Guo et al., 2025; Team, 2025) in the Appendix E, whose behaviors are also consistent with the following observations.

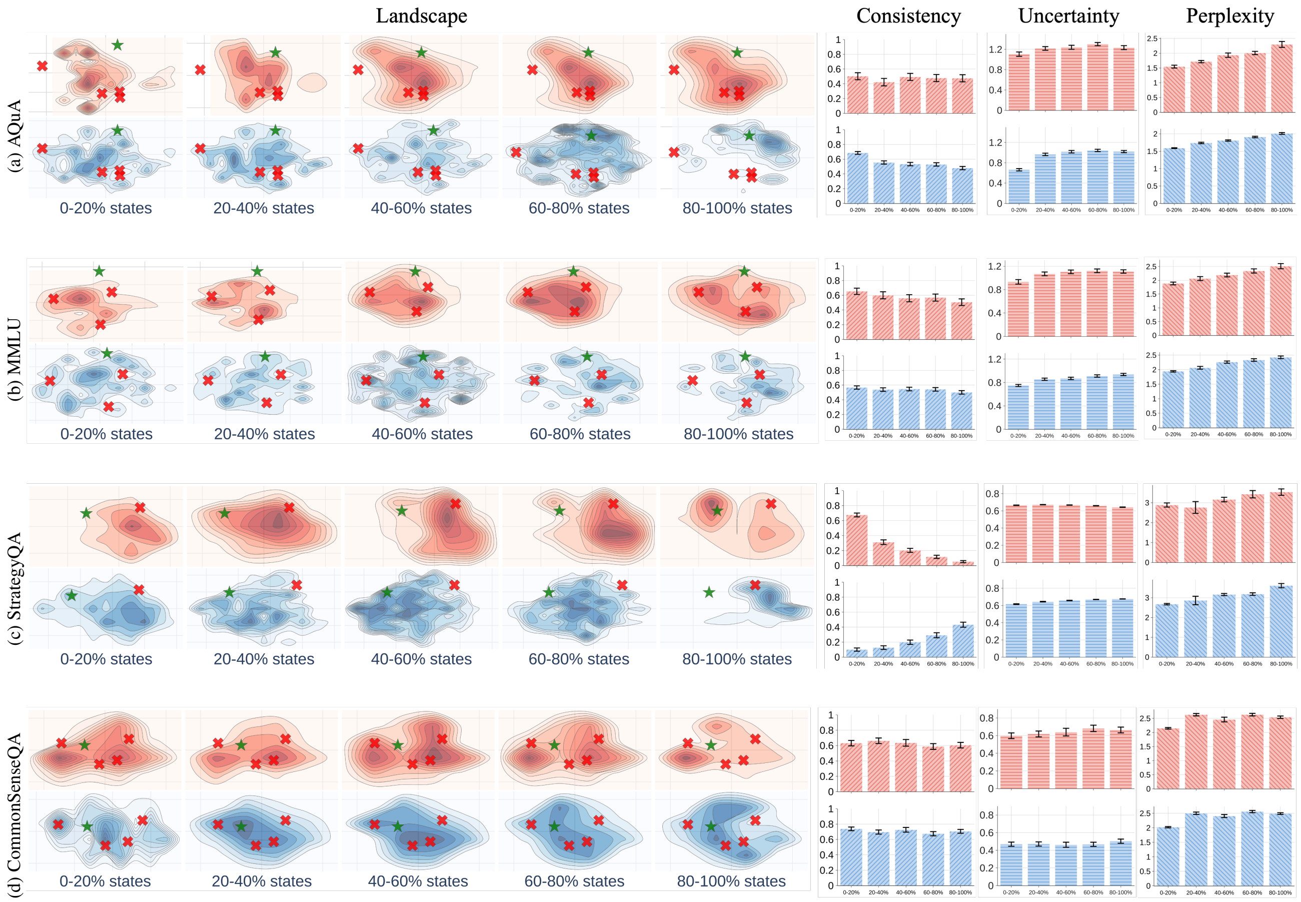


Figure 3: Comparing the landscapes and corresponding metrics of different datasets (using Llama-3.1-70B with CoT). Darker regions represent higher state density, with ✖ indicating incorrect answers and ★ marking the correct ones. In addition, the accuracy of reasoning for the four subfigures is: (a) 84.4%, (b) 80.2%, (c) 75.8%, and (d) 64.8%, respectively.

Observation 3.6 (*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 to store information, larger models can access broader knowledge (Allen-Zhu and Li, 2024). This leads to more confident solutions, demonstrated by more focused answer patterns and lower uncertainty.

Observation 3.7 (*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.

Case study on reasoning model. We visualize the reasoning behavior of the latest reasoning model, namely QwQ-32 B (Team, 2025), via the landscape, shown in Fig. 7. Here, we summarize distinctive reasoning behavior apart from general models as follows:

Observation 3.8 (*Correct reasoning demonstrates decentralized patterns*)**.** By comparing the correct reasoning pattern with those of general models, as shown in Fig. 4, we observe decentralization of landscapes from reasoning models. This is characterized by thoughts distributed broadly across the landscape, indicating that the reasoning model engages in extensive exploration to derive its final decisions.

Observation 3.9 (*Reasoning Models Exhibit Self-Checking and Awareness of Correctness*)**.** By comparing textual representations with reasoning landscapes, we observed that reasoning models promote self-checking behavior when thoughts deviate from the correct answer, often manifesting as 'Aha' moments (Guo et al., 2025). This suggests that reasoning models develop an awareness of the correctness of their reasoning processes.

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

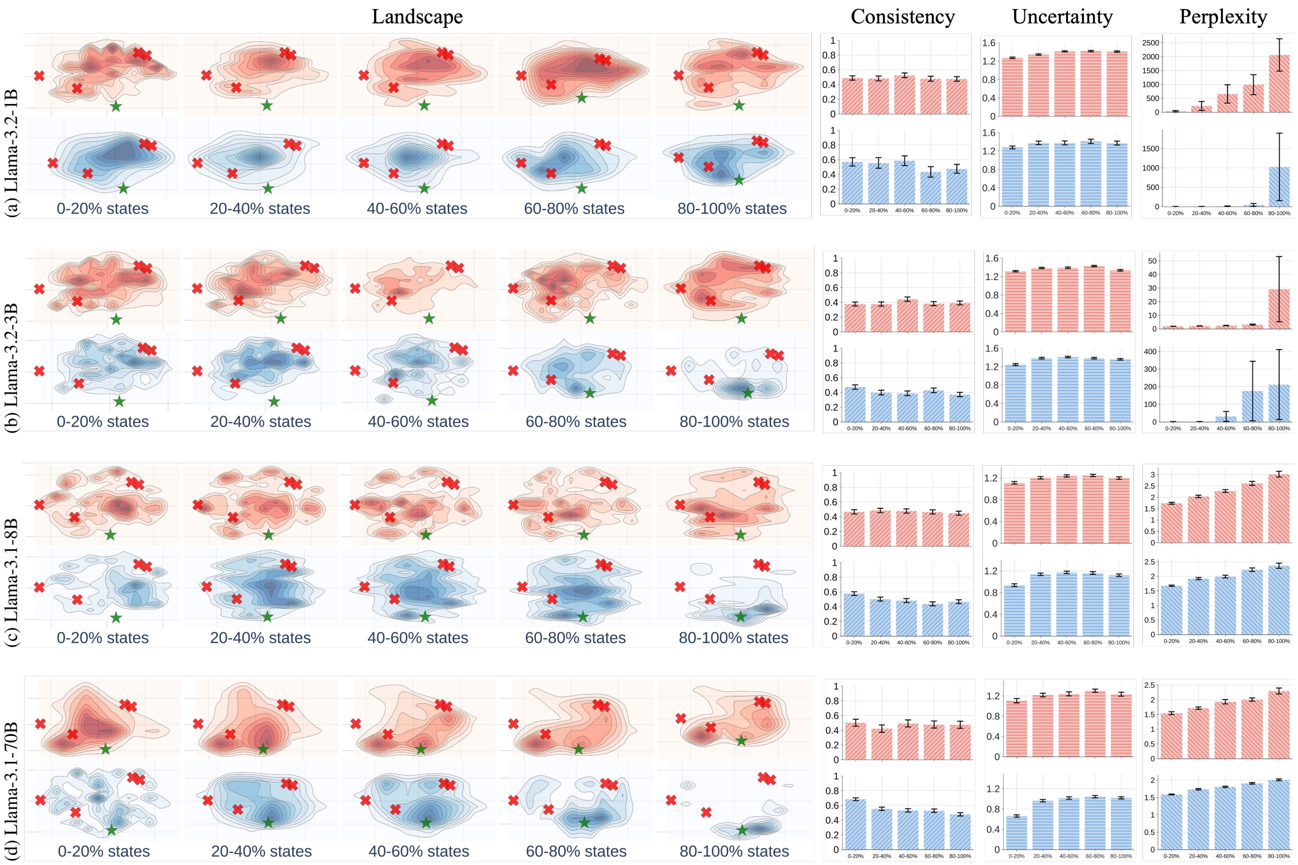


Figure 4: Comparing the landscapes and corresponding metrics of different language models (with CoT on the AQuA dataset). Darker regions represent higher state density, with ✖ indicating incorrect answers and ★ marking the correct ones. In addition, the accuracy of reasoning for the four subfigures is: (a) 15.8%, (b) 42.0%, (c) 53.2%, and (d) 84.4%.

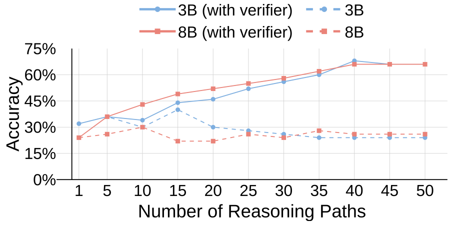
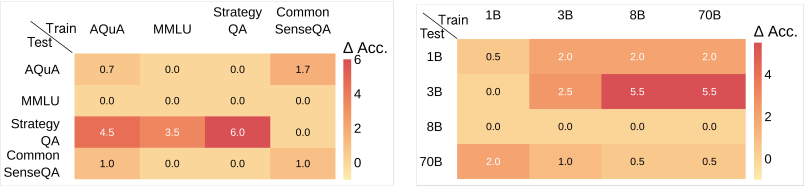


Figure 5: Demonstration of the inference-time scaling effect of the verifier. We show the voting accuracy (%) on StrategyQA scales with the number of reasoning paths.



(a) Transfer across datasets

(b) Transfer across models

Figure 6: Absolute accuracy changes (Δ Acc) with the verifier, compared to performance in Fig. 8 (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.

for voting reasoning paths, 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 3.2 and 3.3 show that the convergence speed and consistency of intermediate states can distinguish correct and wrong paths. Inspired by these observations, we build a model $f : \mathbb{R}^{(k+1)\times n} \rightarrow \{0, 1\}$ to predict the correctness of a reasoning path based on the state features $\{\boldsymbol{s}_i\}_{i=1}^{n}$ and consistency metric $\{\text{Consistency}(s_i)\}_{i=1}^{n}$. 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 (Breiman, 2001) to avoid overfitting. We use this model as a verifier to enhance LLM reasoning (Cobbe et al., 2021). Unlike popular verifiers (Lightman et al., 2023)

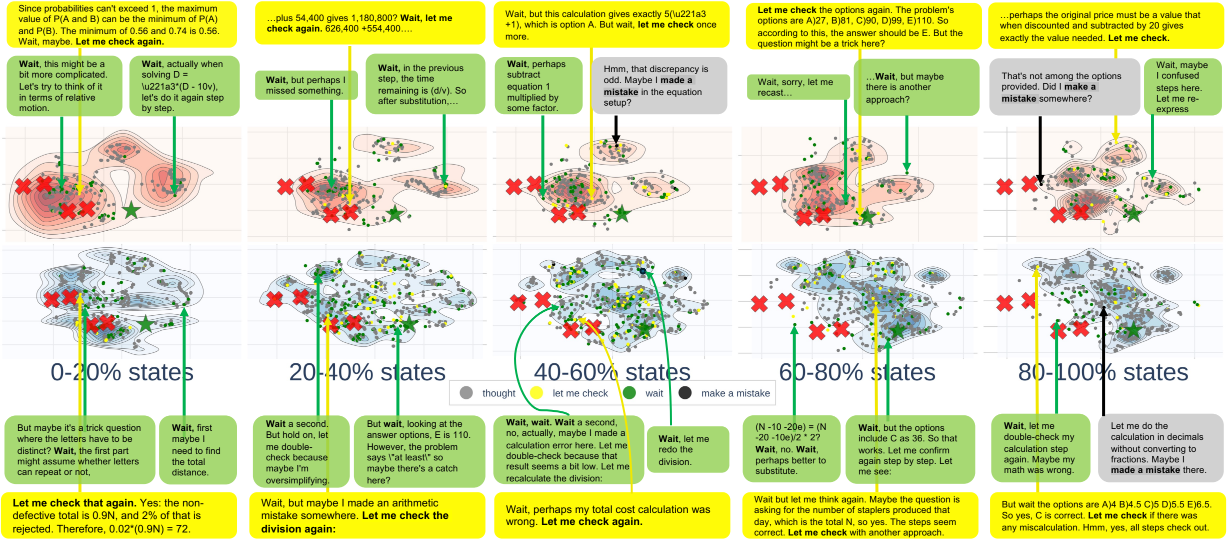


Figure 7: Landscape of QwQ-32B using CoT on AQuA. Better view in color. Distinct reasoning patterns are marked with different colors, including self-evaluation and self-refinement. More visualization can be found at Appendix E.

that involve a moderately sized language model on textual thoughts, our verifier operates on state features and is super lightweight. We train a verifier on thoughts sampled on the training split of each dataset and apply it to vote reasoning paths at test time. Given q paths sampled by a decoding algorithm, the final prediction is produced by a weighted majority voting given by the following equation:

$$\hat{y} = \arg\max_{c \in \mathcal{C}} \sum_{i=1}^{q} \mathbb{1}(\hat{y}^{(i)} = c) \cdot f(\{\boldsymbol{s}_i\}_{i=1}^{n}, \{\text{Consistency}(s_i)\}_{i=1}^{n}). \tag{9}$$

4.2. Experimental Results

We evaluate our numerical verifier against an unweighted voting baseline (Wang et al., 2023b) with various models, decoding algorithms, and reasoning datasets. Detailed settings and results are in Appendix C.1.

Effectiveness of the verifier. We first compare our verifier against the unweighted voting baseline, each applied to 10 reasoning paths. As shown in Fig. 8, our verifier consistently enhances the reasoning performance of all models and decoding algorithms, even though our verifier does not use any pre-trained language model. Notably, smaller language models (1B and 3B) show significant performance gains with the verifier's assistance, achieving substantial improvements over their original capabilities of reasoning. We also compare the verifier between reward-guided algorithms

Test-time scaling. While the improvement of the verifier seems marginal with 10 reasoning paths, our verifier can provide a substantial performance gain with more reasoning paths. We adjust the number of reasoning paths from 1 to 50, and plot the results of the verifier and the unweighted voting baseline in Fig. 5. 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. 6, 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. Related Work

Reasoning with large language models. Chain-of-Thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) 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

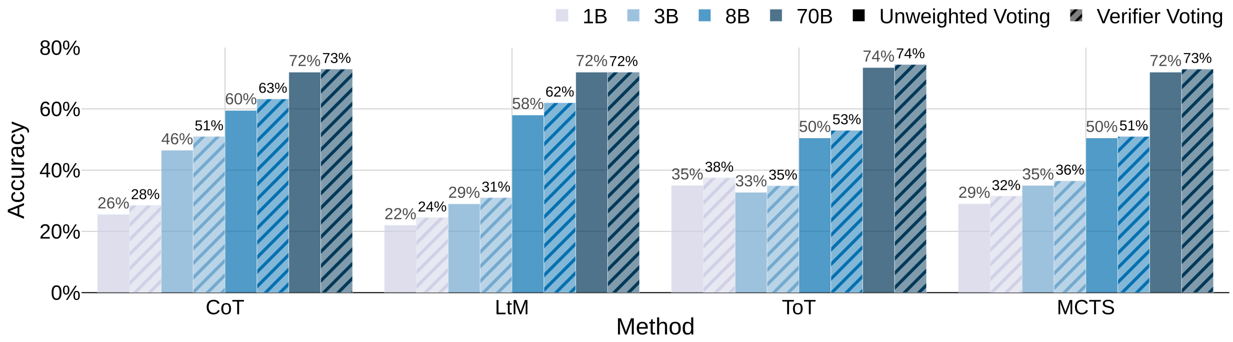


Figure 8: Accuracy under methods and model scales (averaging across four datasets). Dataset-level results see Appendix E.

proposed to address various challenges, including compositional generalization (Zhou et al., 2023; Khot et al., 2023), planning (Yao et al., 2023a; Hao et al., 2023), and rule learning (Zhu et al., 2023) within the CoT reasoning. Beyond solving reasoning tasks, CoT has also emerged as a foundational framework for other techniques, such as fine-tuning LLMs (Zelikman et al., 2022), enabling LLM-based agents (Yao et al., 2023b), and facilitating test-time scaling (Snell et al., 2024). 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 (2022) found that the text and patterns of exemplars help CoT generate sentences resembling correct answers. Besides, Wang et al. (2023a) highlighted the importance of maintaining the correct order of reasoning steps, while Ye et al. (2022) demonstrated that using complementary exemplars can enhance reasoning performance. Furthermore, CoT can benefit from longer reasoning chains, even without new information to the prompt (Jin et al., 2024). Another line of research investigates CoT's general behavior (Tang et al., 2023; Saparov and He, 2023; Saparov et al., 2023; Shi et al., 2023). For example, CoT heavily depends on the semantic structure of the problem to perform reasoning (Tang et al., 2023), struggles with planning and unification in deductive reasoning (Saparov and He, 2023), has difficulty generalizing to longer reasoning paths (Saparov et al., 2023), and can be easily misled by irrelevant information in the context (Shi et al., 2023). 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 (Wu et al., 2023), 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 (Wu et al., 2023) is that CoT stabilizes saliency scores on semantically relevant tokens compared to direct prompting. Metrics designed to quantify CoT performance (Chen et al., 2024; Ton et al., 2024) can also be used to analyze the reasoning behaviors of LLMs. For instance, Ton et al. (2024) employs information gain to identify failure modes in reasoning paths, aligning with Observation 3.2 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 (Cobbe et al., 2021).

6. Conclusion

This paper introduces the landscape of thoughts, a visualization tool for analyzing the reasoning paths produced by large language models with chain-of-thought. Built on top of feature vectors of intermediate states in reasoning paths, 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 observed property, which is demonstrated by a lightweight verifier developed based on the feature vectors and our observations. We foresee that this tool will create several opportunities to develop, understand, and monitor the LLM reasoning.

One limitation of the landscape of thoughts is its applicability only to multiple-choice tasks. Future work could focus on adapting this tool for open-ended reasoning tasks, such as mathematical problem-solving, code generation, and planning, where reasoning paths are less structured and more complex. Additionally, further research could aim to make the tool more accessible by generating intuitive visual and textual explanations, enabling non-experts to better understand and trust the reasoning processes of LLMs. Another promising direction is the development of automated methods to detect reasoning failures at scale, which could enhance the reliability of LLMs across diverse applications.

References

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.

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

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

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.

Leo Breiman. Random forests. *Machine learning*, 2001.

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.

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.

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.

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

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.

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.

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.

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.

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.

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.

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

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

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.

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.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

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.

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

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.

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

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

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.

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.

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.

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.

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.

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

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

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

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.

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

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.

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.

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

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.

Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

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.

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

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

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.

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.

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

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

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

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

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*, 2023a.

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.

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*, 2023b.

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.

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.

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.

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*, 2023a.

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*, 2023b.

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

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.

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

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.

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.

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

In this section, we further discuss the challenges in developing the system for analyzing LLMs' reasoning (Appendix B.1), followed by comparing the proposed landscape visualization technique with the textual analysis methodology (Appendix B.2). In addition, we compare the lightweight verifier to conventional reward-guided algorithms (Appendix B.3).

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 (Clark et al., 2019; Kobayashi et al., 2020), probing (Alain and Bengio, 2016; Tenney et al., 2019; Hewitt and Liang, 2019), and circuits (Elhage et al., 2021; Yao et al., 2024), 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 (Ippolito et al., 2022; Wang et al., 2024). 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 of individual samples, as their responses are interpretable by humans. However, this approach has two major limitations:

Limitation 1: Lack of Scalability. Analyzing individual samples 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 sample, reviewing 100 samples 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 like accuracy when experimenting with changes to models and methods.

Limitation 2: Lack of Aggregation. It is difficult to aggregate insights across multiple samples to understand model behavior at the dataset level. Summarizing model behavior across multiple samples 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 (van der Maaten and Hinton, 2008), the visualization enables a more comprehensive analysis of multiple reasoning samples instead of only one sample. 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. A Comparison Between Lightweight Verifier and Reward-guided Algorithms

It is worth noting to mention 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) (Luo et al., 2024; Xu et al., 2025), 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 a 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. Experiment Settings

C.1. Settings

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 (Wang et al., 2023b).

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 samples 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 samples, as introduced in the paragraph below Equation 4. We then aggregate feature vectors from all samples 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.

C.2. Datasets

AQuA (Ling et al., 2017). 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 (Hendrycks et al., 2021). 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 (Geva et al., 2021). 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 (Talmor et al., 2019). This dataset assesses commonsense reasoning through multi-choice questions derived from the ConceptNet knowledge graph (Speer et al., 2017). 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.

C.3. Decoding Algorithms

Chain of Thought (CoT) (Wei et al., 2022). 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 (Kojima et al., 2022). The core idea of this prompt strategy lies in adding simple instructions, e.g., "Let's

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

(a) Verifying Obs. 3.1

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

(b) Verifying Obs. 3.2 and 3.6

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

(c) Verifying Obs. 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

think step by step." to the prompt, enabling models to generate reasoning traces without assigned task-specific examples.

Least-to-Most (LtM) (Zhou et al., 2023). 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) (Yao et al., 2023a). 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) (Zhang et al., 2024). 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 (Zhang et al., 2024) to conduct our experiment across different tasks.

D. Supplementary Results and Analysis

D.1. Statistical Verification of the Observations

In this part, we conduct extra experiments and statistically verify Obs. 3.1, 3.2, 3.4, and 3.6, while the other Obs. 3.3, 3.5, and 3.7 have been quantitatively verified by the metrics in Sec. 2.3.

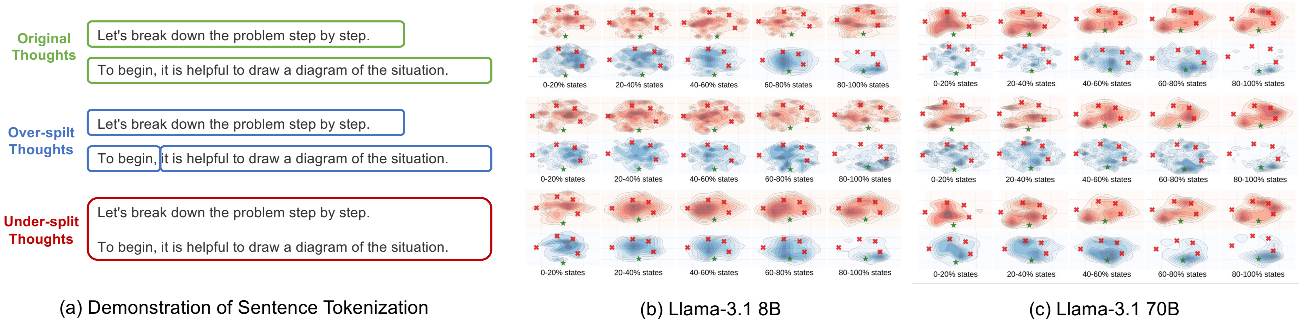
To verify Obs. 3.1, we calculate the convergence coefficient (e^β) by fitting a log-linear regression model to the sequence of distances d_i between each state and the final answer as $\log(d_i) \approx \alpha + \beta i$, where α is the intercept term; β is the slope coefficient that quantifies convergence behavior; i represents the position index in the reasoning chain. Lower values of e^β indicate faster convergence. For Obs. 3.2 and 3.6, 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 Obs. 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 bs. 3.1. Shown in Tab. 1(b), speed and accuracy correlate strongly (p-value = 9.421e-11), thus verifying Obs. 3.2. This is also applicable for verifying Obs. 3.6. Tab. 1(c) shows that lower scores indicate greater dissimilarity between landscapes, which verifies Obs. 3.4, i.e., AQuA, MMLU, and StrategyQA are more similar, while CommonSenseQA exhibits distinct patterns.

D.2. 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 the controlled experiment by considering two imperfect cases in thought split, namely over-split thoughts and under-split thoughts.

Specifically, shown as Fig. 9 (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. 2(a) and Fig. 4(c),



(a) Demonstration of Sentence Tokenization

(b) Llama-3.1 8B

(c) Llama-3.1 70B

Figure 9: Demonstration of sentence tokenization methods for thoughts splitting.

Table 2: Absolute accuracy with the verifier, compared to performance in Fig. 8 (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)

Shown in Fig. 9 (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.

D.3. Absolute Performance of the Verifier

In this part, we provide the absolute performance of the experiment conducted in Fig. 6. Shown as Tab. 2, the results demonstrate that our approach consistently provides improvements across different domains and models.

D.4. 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 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. 3 show that this process verifier is comparable to the outcome verifier.

D.5. Further Discussion on the StrategyQA

The abnormal reasoning behavior, where states cluster on anchors that differ from their final answer in Fig. 3(c), 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.

Table 3: 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

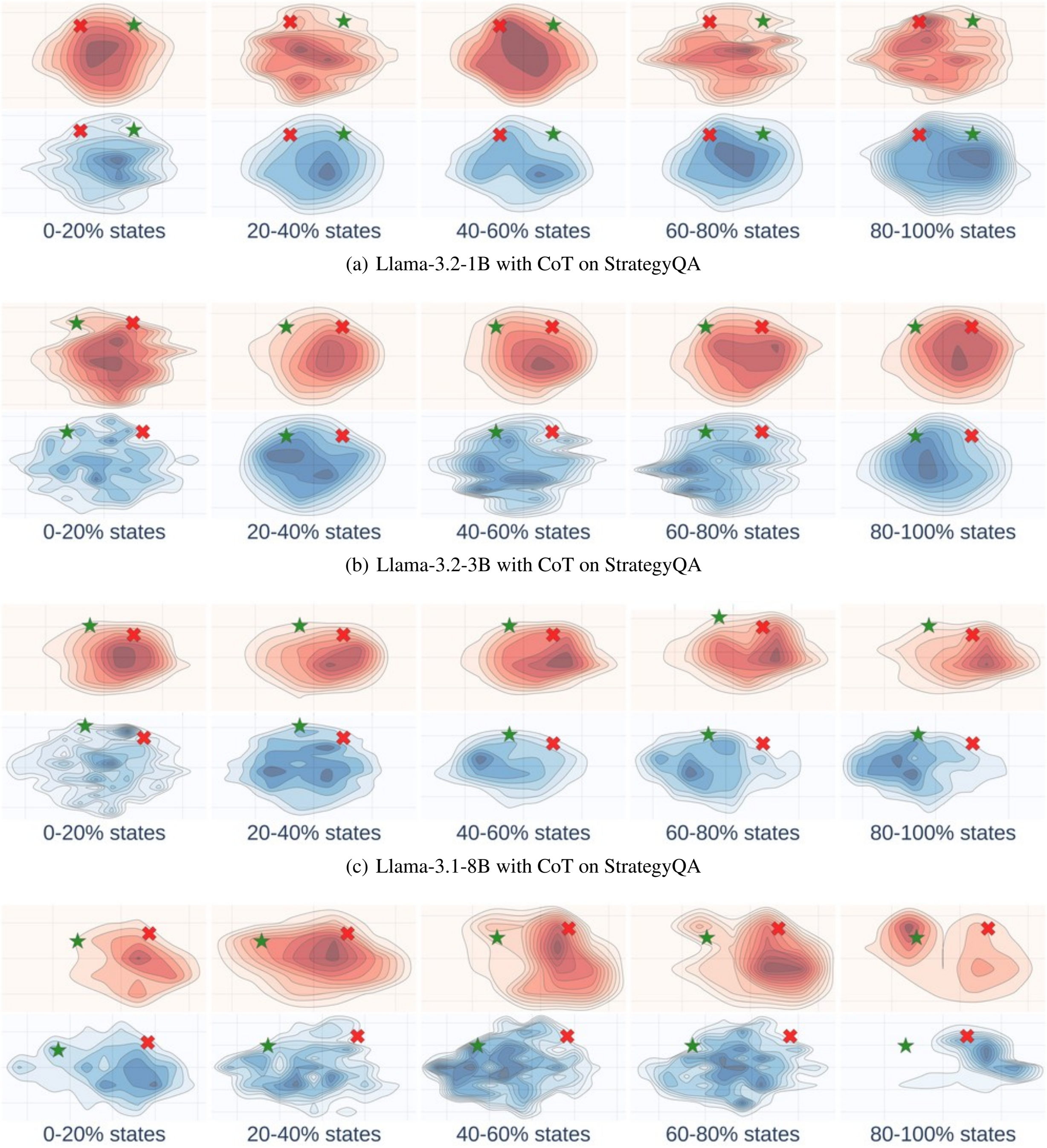
Further, we provide landscape visualizations for the same dataset using other models and methods in Fig. 10 to Fig. 13. These landscapes do not exhibit the same abnormal density patterns, reinforcing that the issue is specific to Llama-3.1-70B's reasoning instability rather than a flaw in our visualization framework.

E. Visulizations

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

In Fig. 14 to Fig. 17, we visualize the average voting accuracy (%) of different LLMs reasoning with and without verification on various datasets and methods. In Fig. 18 to Fig. 21, 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 22 to Fig. 25.

In addition, we provide the landscape of thoughts on the latest reasoning model. Specifically, we conduct additional experiments on the DeepSeek-R1-Distill model (Guo et al., 2025) (Llama-70 B and Qwen-1.5 B). As shown in Fig. 26 and Fig. 27, 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.



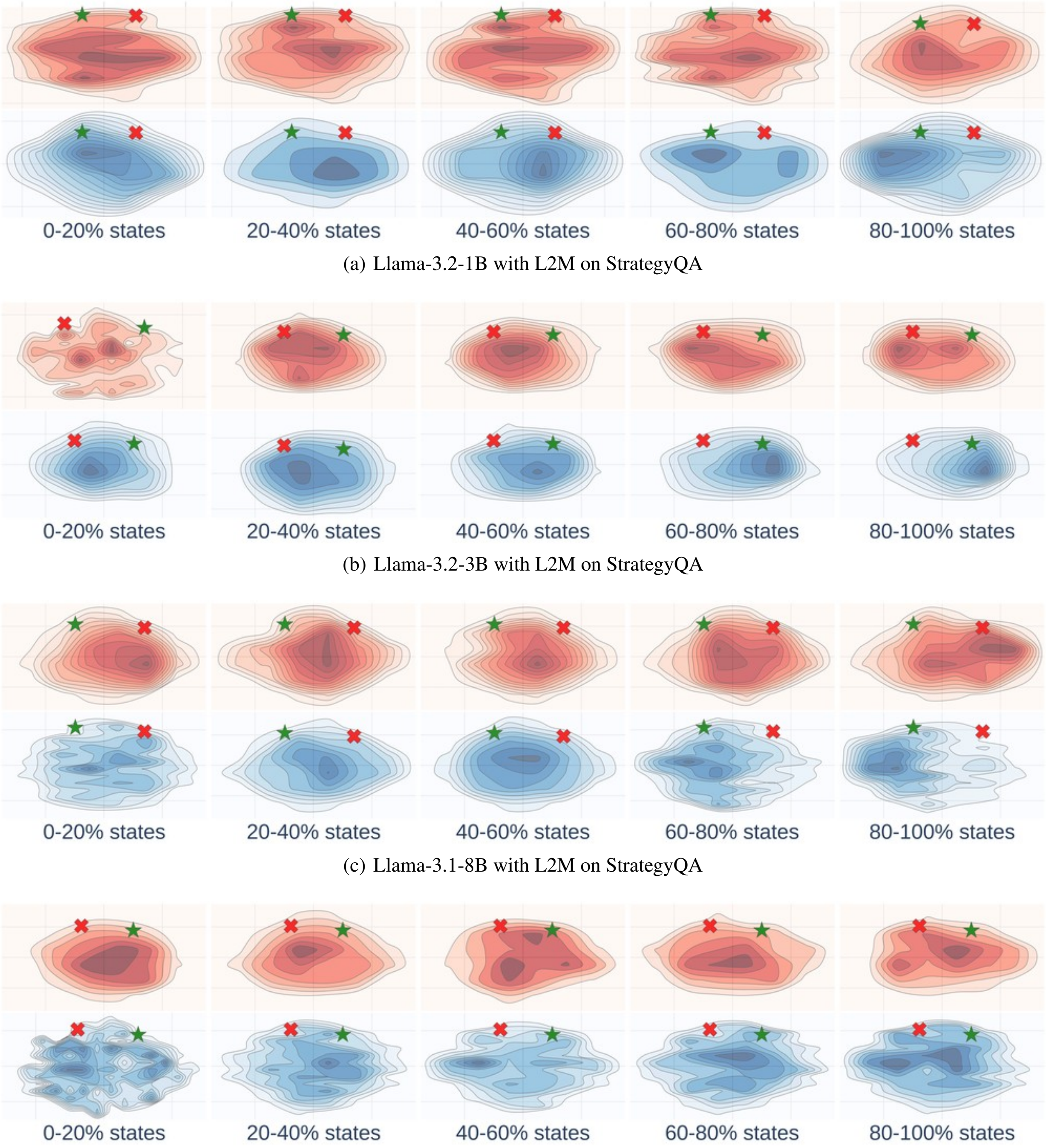
(a) Llama-3.2-1B with CoT on StrategyQA

(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).



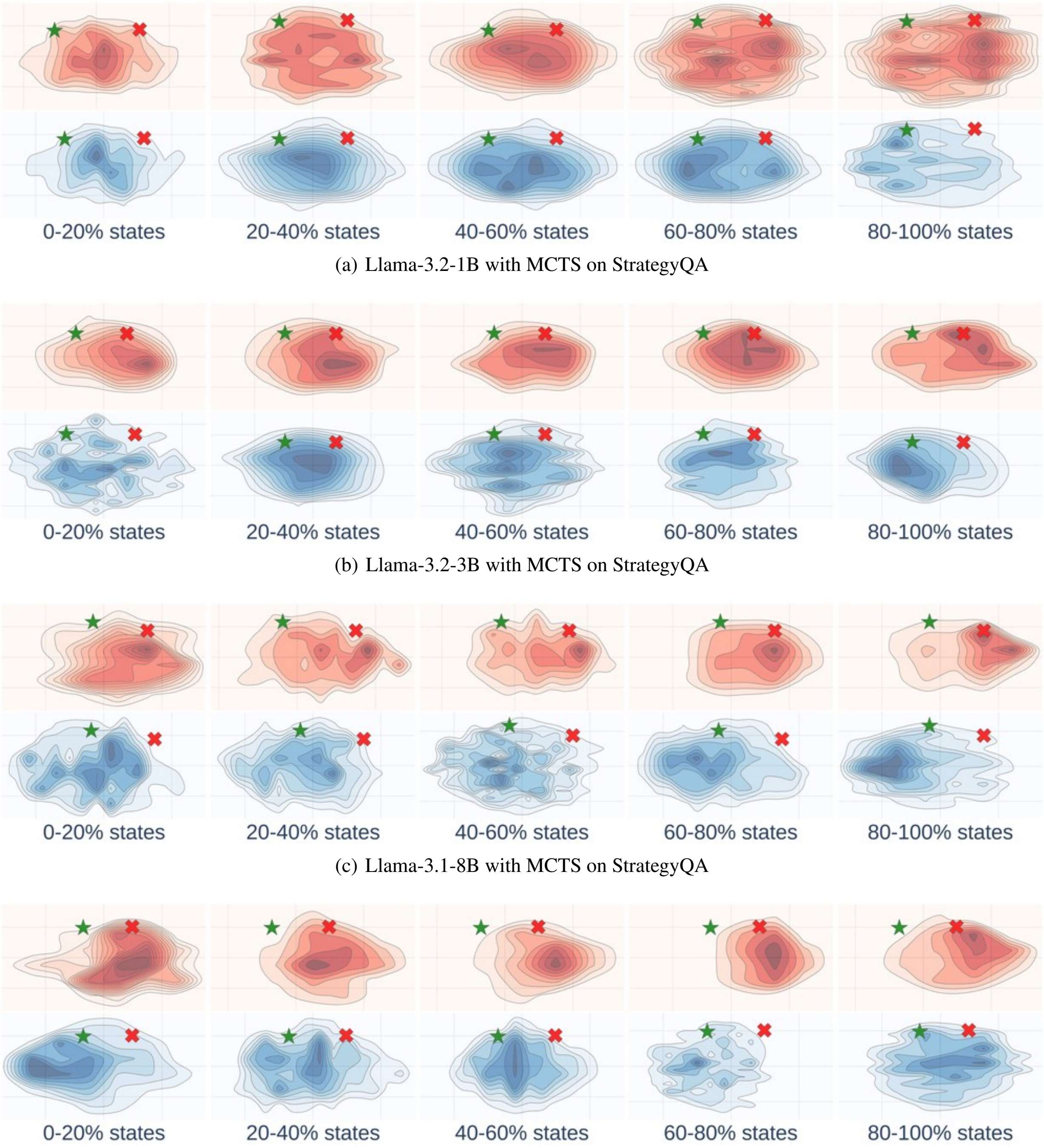
(a) Llama-3.2-1B with L2M on StrategyQA

(b) Llama-3.2-3B with L2M on StrategyQA

(c) Llama-3.1-8B with L2M on StrategyQA

(d) Llama-3.1-70B with L2M on StrategyQA

Figure 11: The landscapes of the model across scales (using L2M on the StrategyQA dataset).



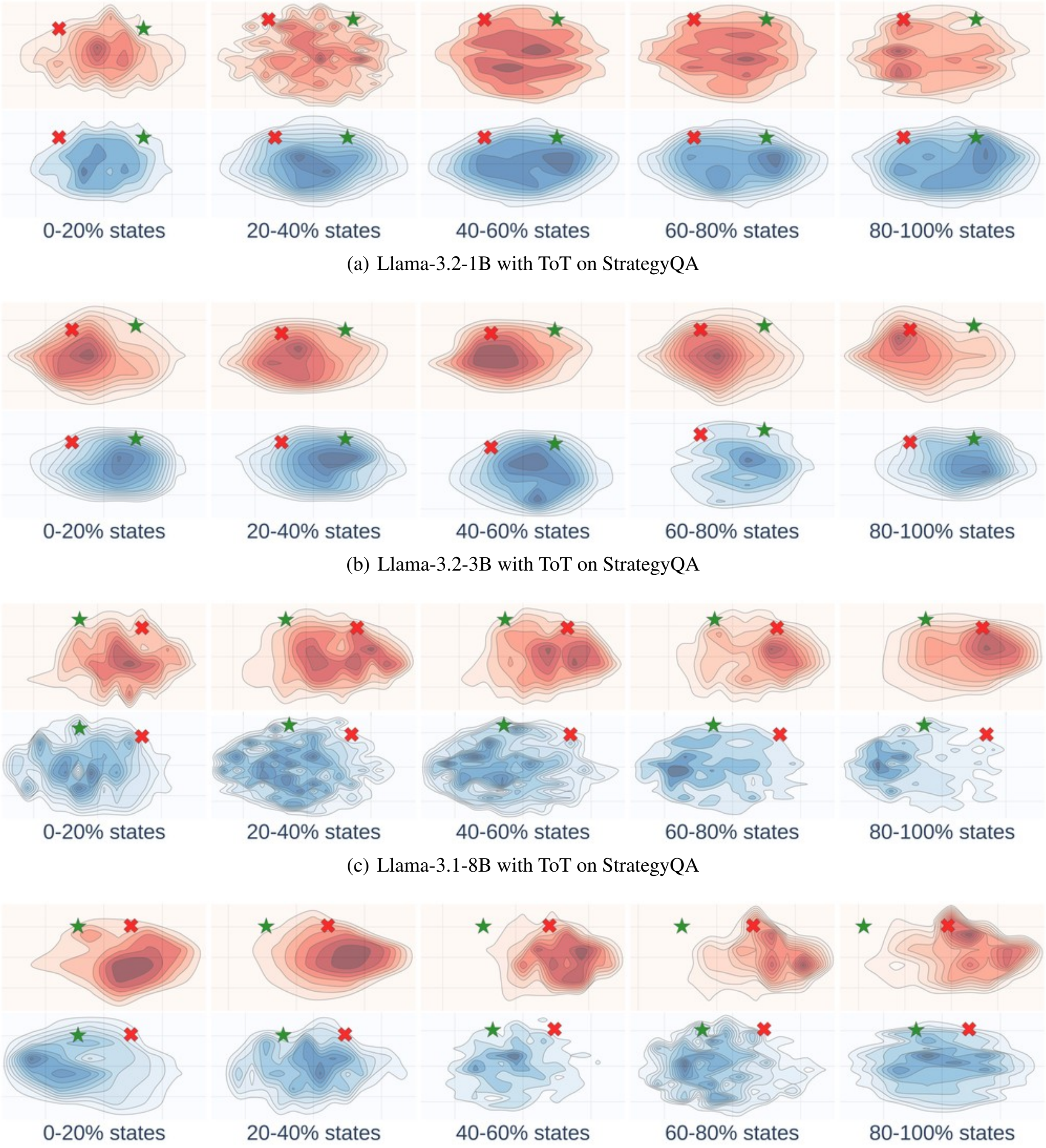
(a) Llama-3.2-1B with MCTS on StrategyQA

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



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

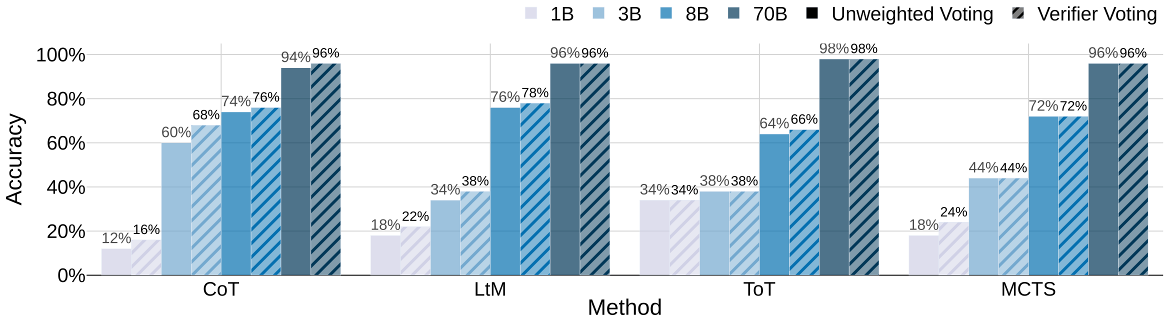


Figure 14: Average voting accuracy (%) of reasoning with and without verification on AQuA.

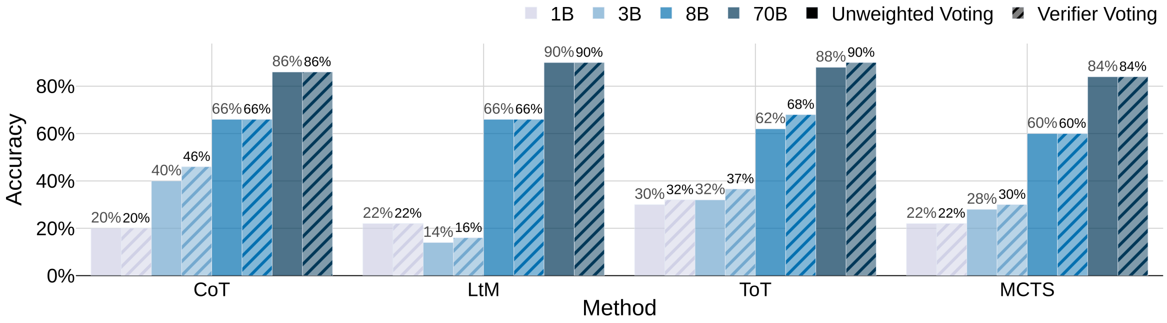


Figure 15: Average voting accuracy (%) of reasoning with and without verification on MMLU.

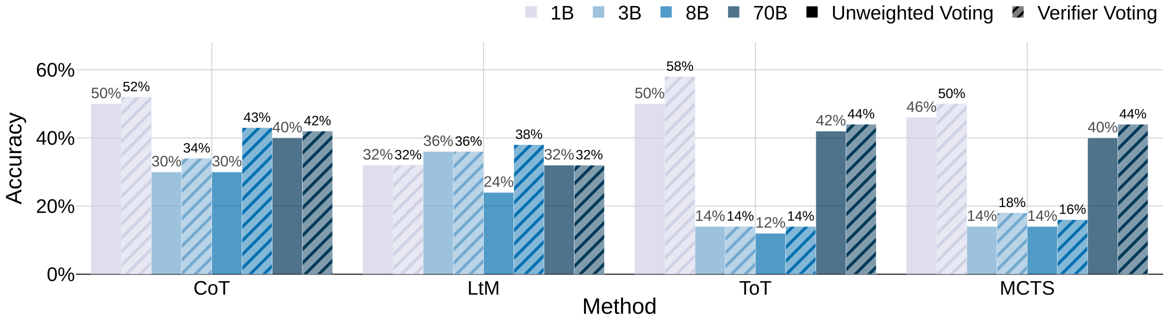


Figure 16: Average voting accuracy (%) of reasoning with and without verification on StrategyQA.

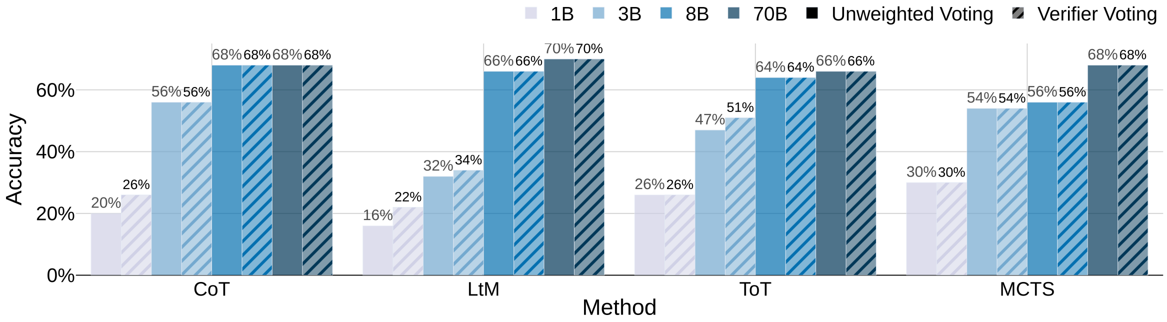
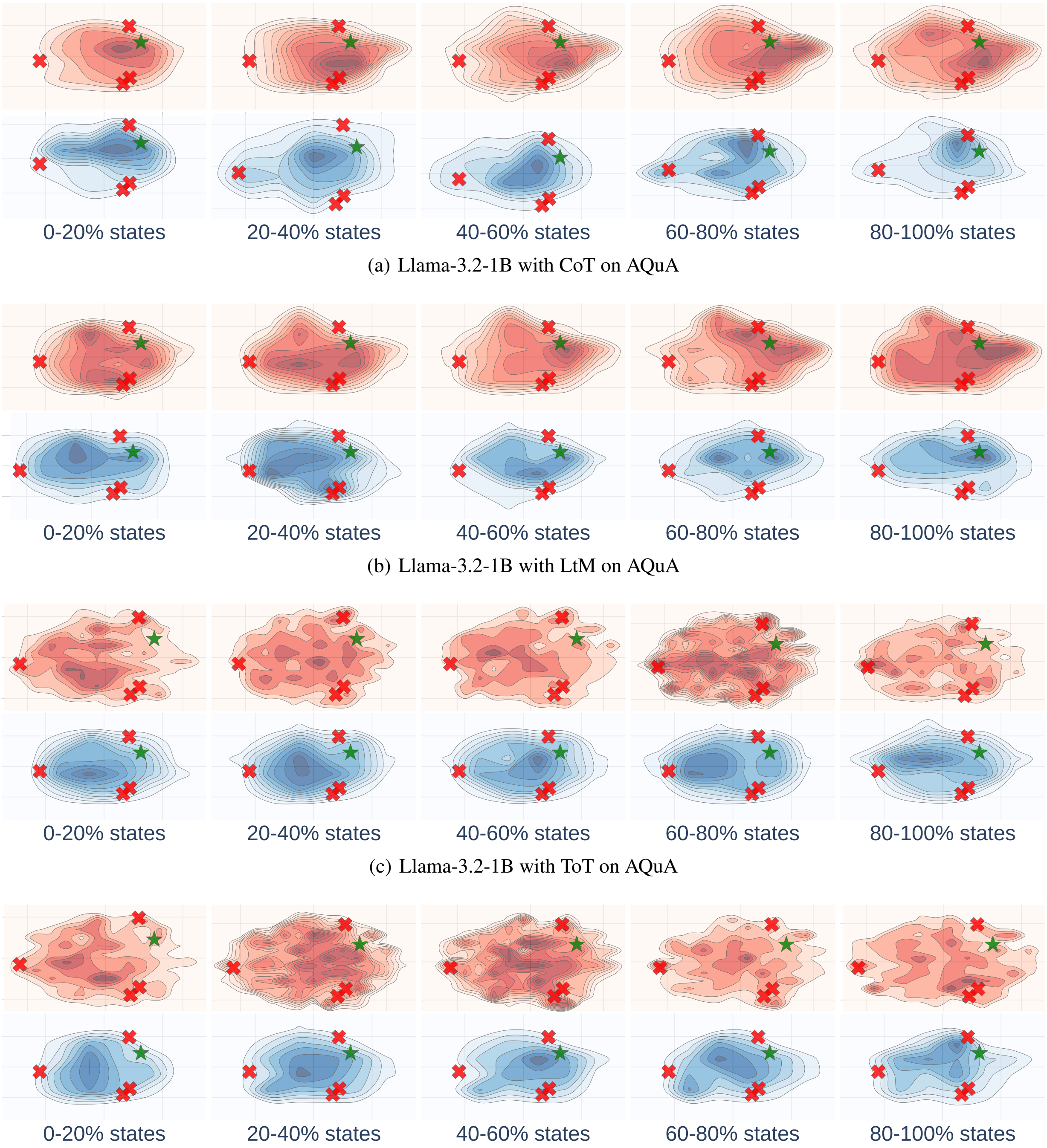


Figure 17: Average voting accuracy (%) of reasoning with and without verification on CommonSenseQA.



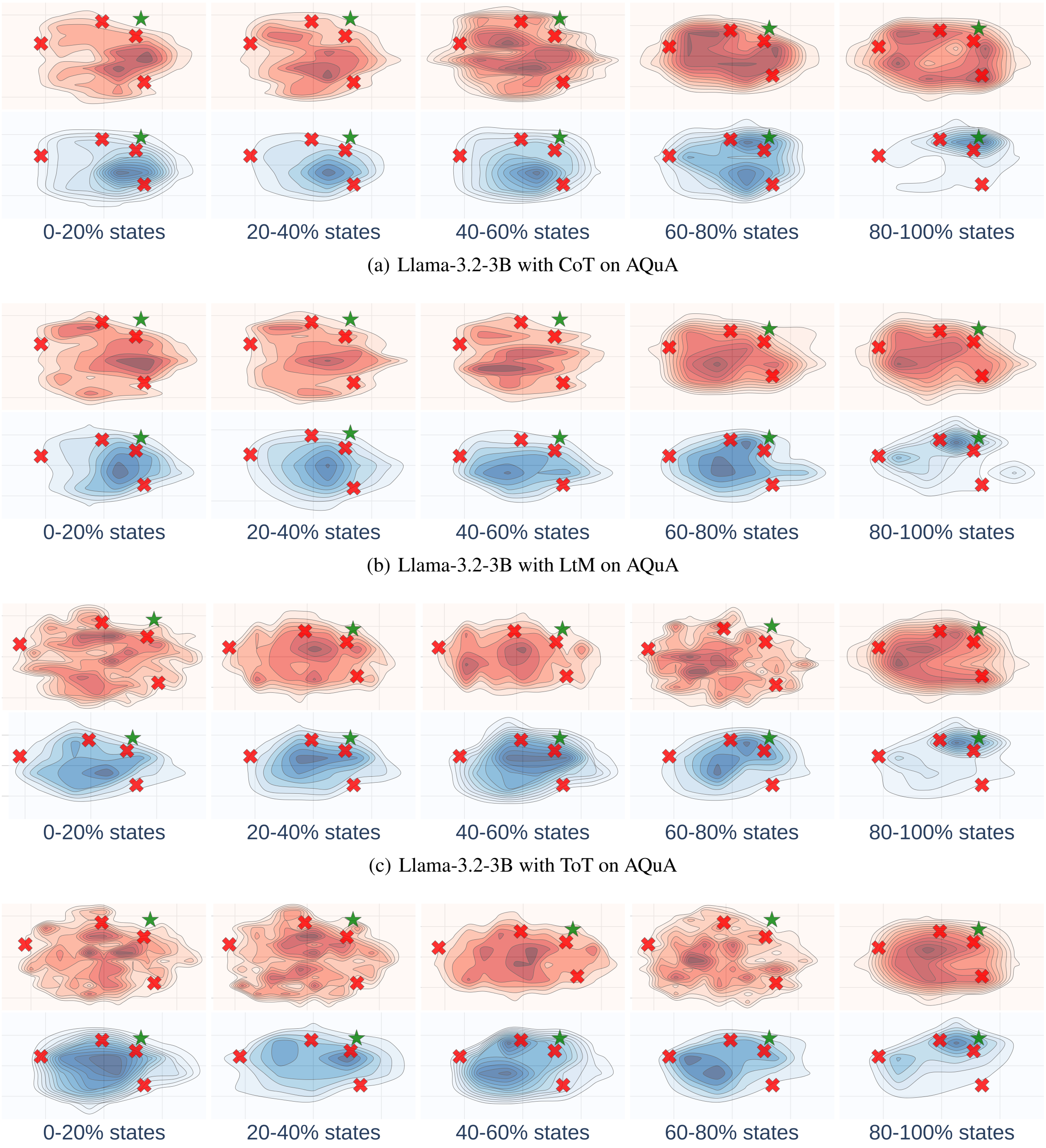
(a) Llama-3.2-1B with CoT on AQuA

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



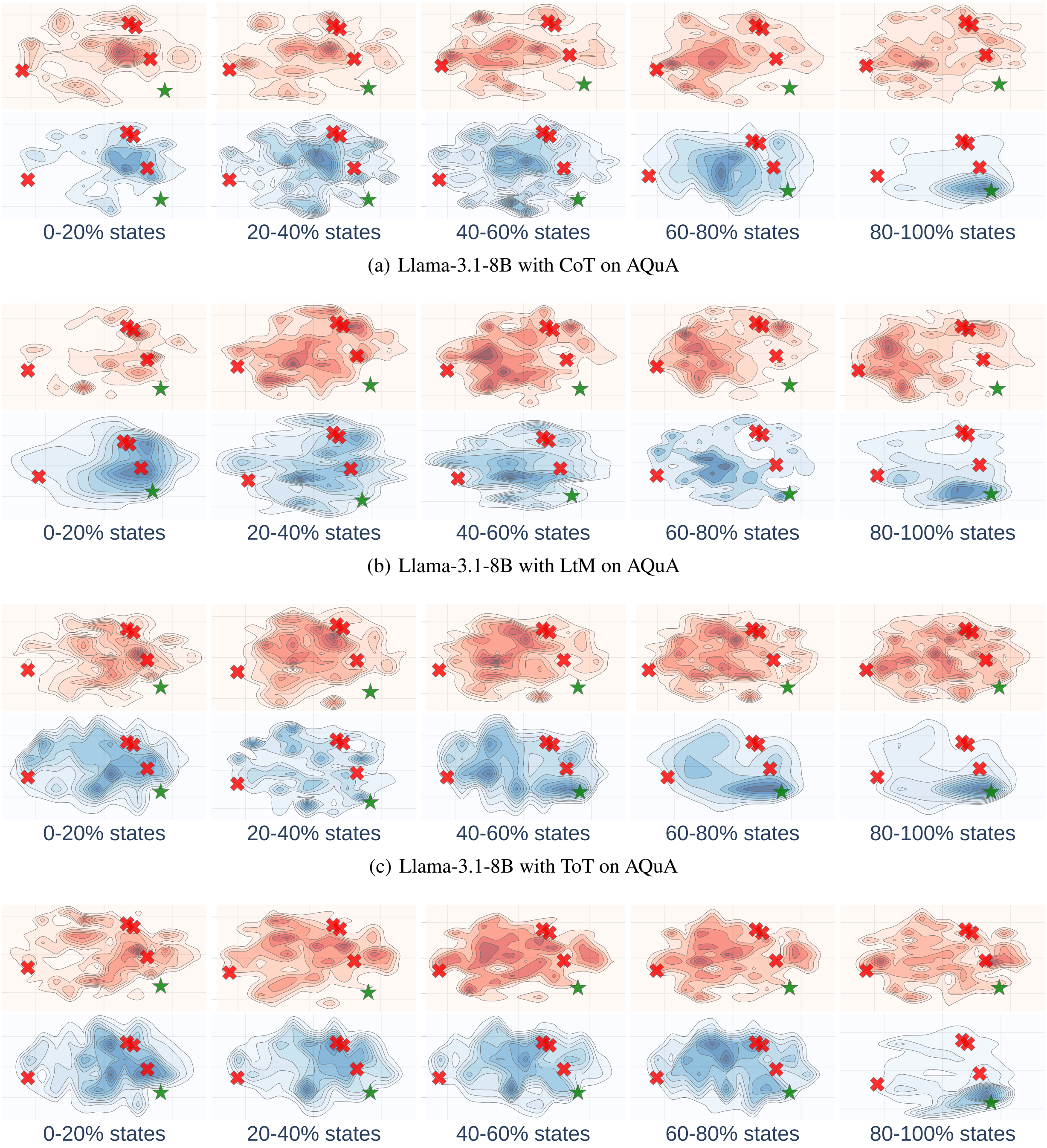
(a) Llama-3.2-3B with CoT on AQuA

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



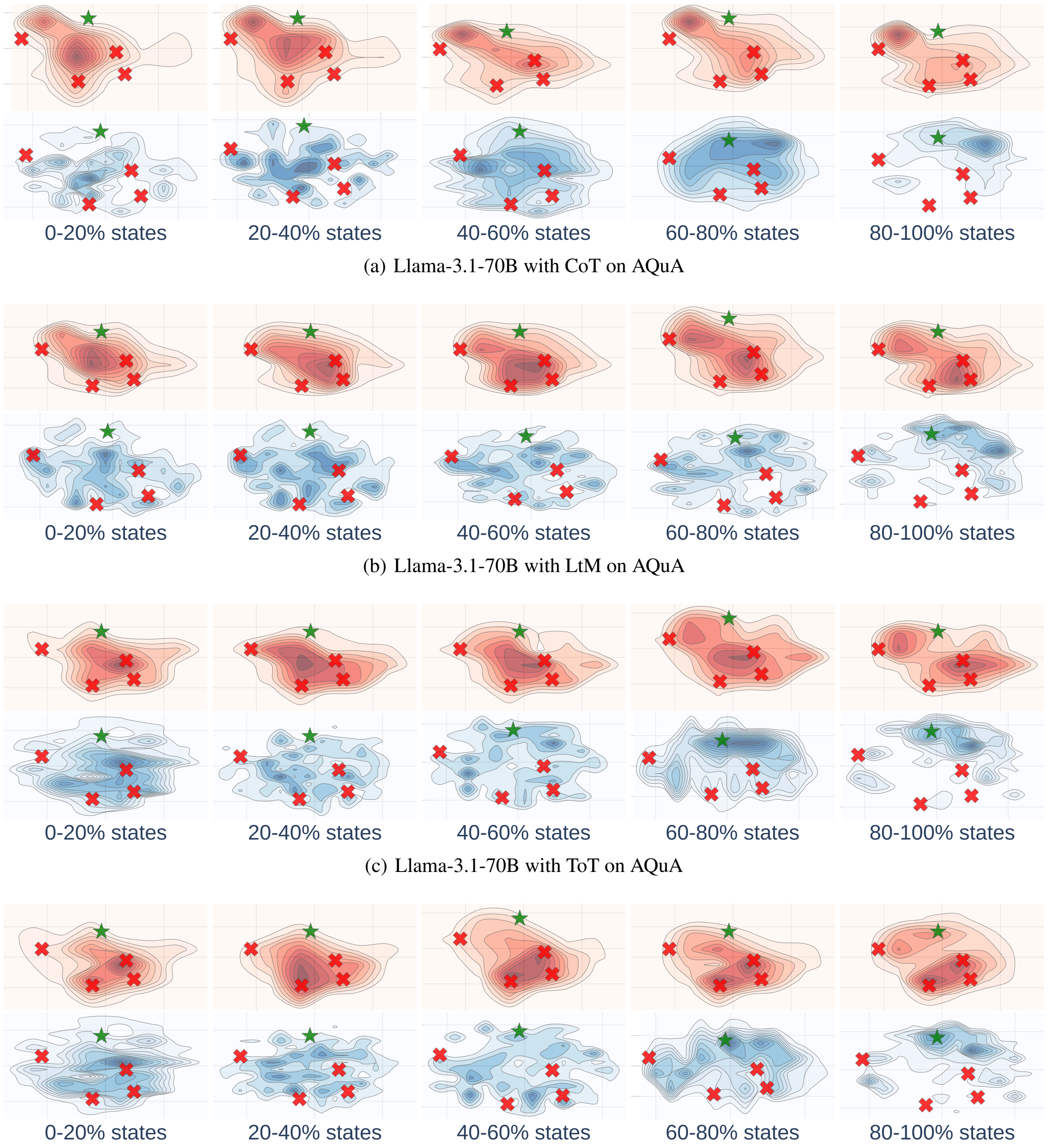
(a) Llama-3.1-8B with CoT on AQuA

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



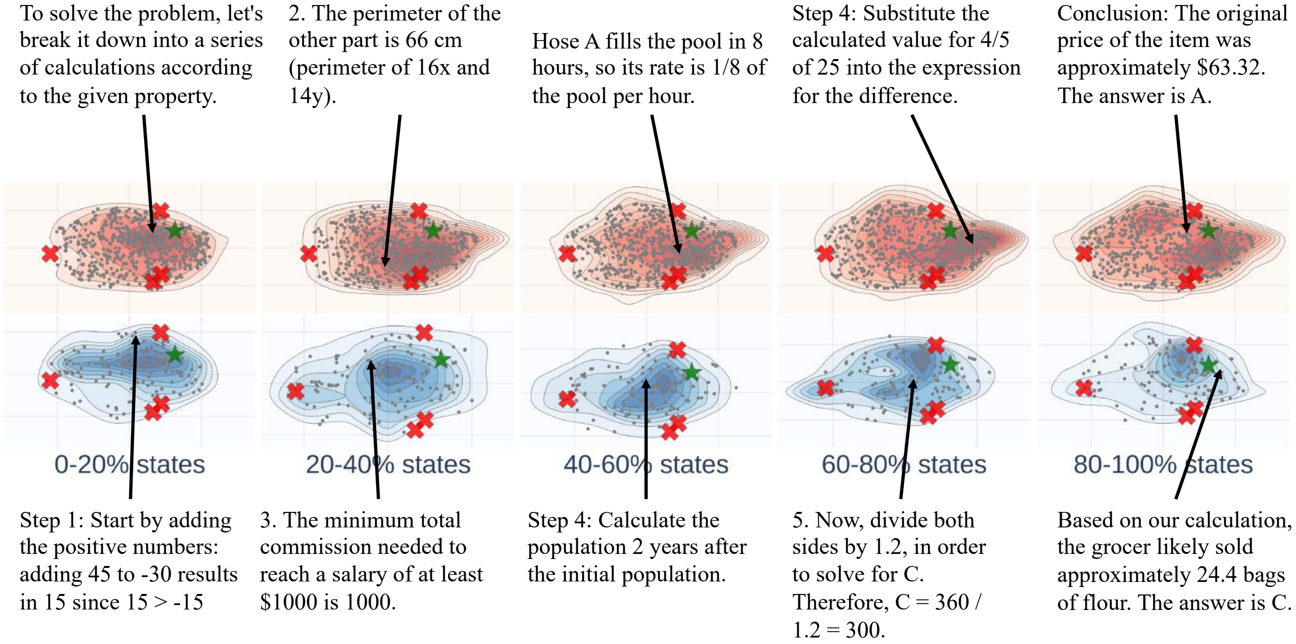
(a) Llama-3.1-70B with CoT on AQuA

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



To solve the problem, let's break it down into a series of calculations according to the given property.

2. The perimeter of the other part is 66 cm (perimeter of 16x and 14y).

Hose A fills the pool in 8 hours, so its rate is 1/8 of the pool per hour.

Step 4: Substitute the calculated value for 4/5 of 25 into the expression for the difference.

Conclusion: The original price of the item was approximately $63.32. The answer is A.

Step 1: Start by adding the positive numbers: adding 45 to -30 results in 15 since 15 > -15

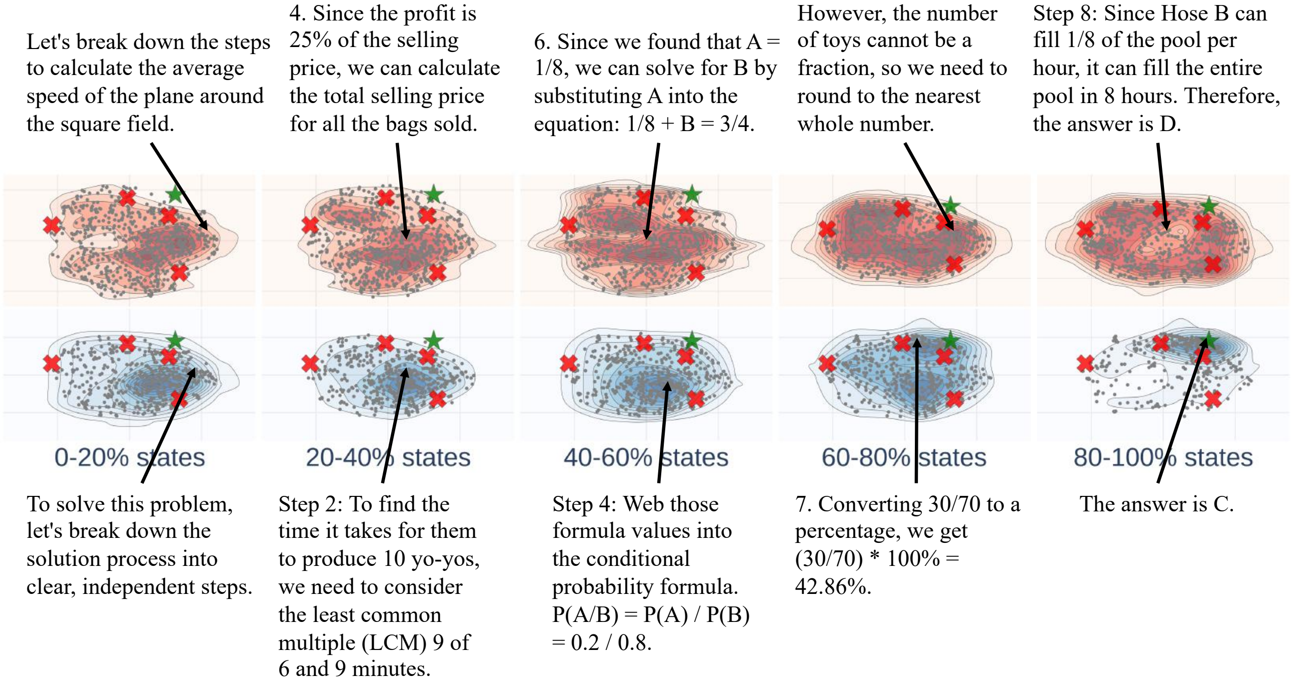
3. The minimum total commission needed to reach a salary of at least $1000 is 1000.

Step 4: Calculate the population 2 years after the initial population.

5. Now, divide both sides by 1.2, in order to solve for C. Therefore, C = 360 / 1.2 = 300.

Based on our calculation, the grocer likely sold approximately 24.4 bags of flour. The answer is C.

Figure 22: Case Study: Landscape of thoughts of Llama-3.2-1B on AQuA using CoT.



Let's break down the steps to calculate the average speed of the plane around the square field.

4. Since the profit is 25% of the selling price, we can calculate the total selling price for all the bags sold.

6. Since we found that A = 1/8, we can solve for B by substituting A into the equation: 1/8 + B = 3/4.

However, the number of toys cannot be a fraction, so we need to round to the nearest whole number.

Step 8: Since Hose B can fill 1/8 of the pool per hour, it can fill the entire pool in 8 hours. Therefore, the answer is D.

To solve this problem, let's break down the solution process into clear, independent steps.

Step 2: To find the time it takes for them to produce 10 yo-yos, we need to consider the least common multiple (LCM) 9 of 6 and 9 minutes.

Step 4: Web those formula values into the conditional probability formula. P(A/B) = P(A) / P(B) = 0.2 / 0.8.

7. Converting 30/70 to a percentage, we get (30/70) * 100% = 42.86%.

The answer is C.

Figure 23: Case Study: Landscape of thoughts of Llama-3.2-3B on AQuA using CoT.

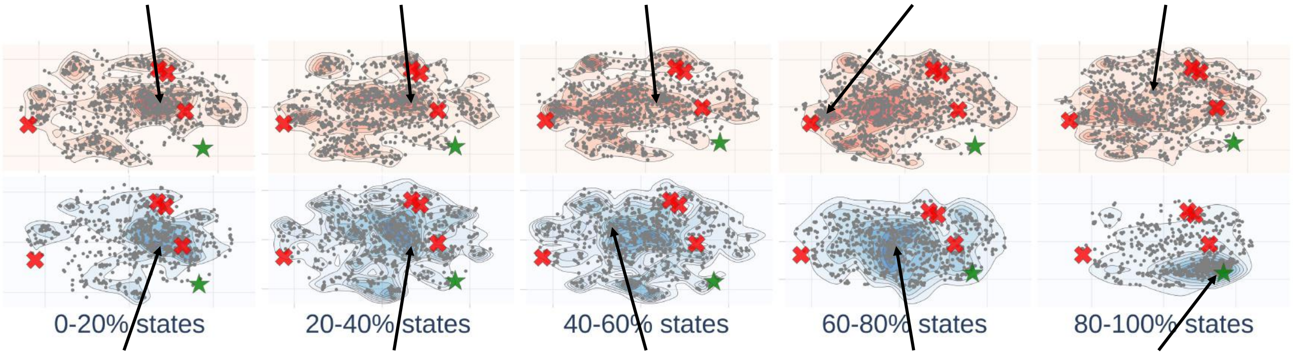
Let's think step by step to solve this problem.

Step 3: The total cost can be expressed as the sum of costs of brown and white sharpeners: b X + (18 - b) (× + 1) = 100.

Next, we divide the total profit by the profit per bag: $3,000 / $25 = 120.

Step 8: Solve for x using the equation identified in step 7.

Conclusion: The original price of the item was approximately $63.32. The answer is A



Let‘s think step by step to solve the problem.

3. This leaves 1 1/4 = 3/4 of the pool to be filled by both hoses working together in the following 3 hours.

We can represent the commission of 15% on the monthly sale as: 0.15 * total monthly sale.

To find the percentage increase, we'll use the formula: ((Increase / Original) 100). *

The answer is B.

Figure 24: Case Study: Landscape of thoughts of Llama-3.1-8B on AQuA using CoT.

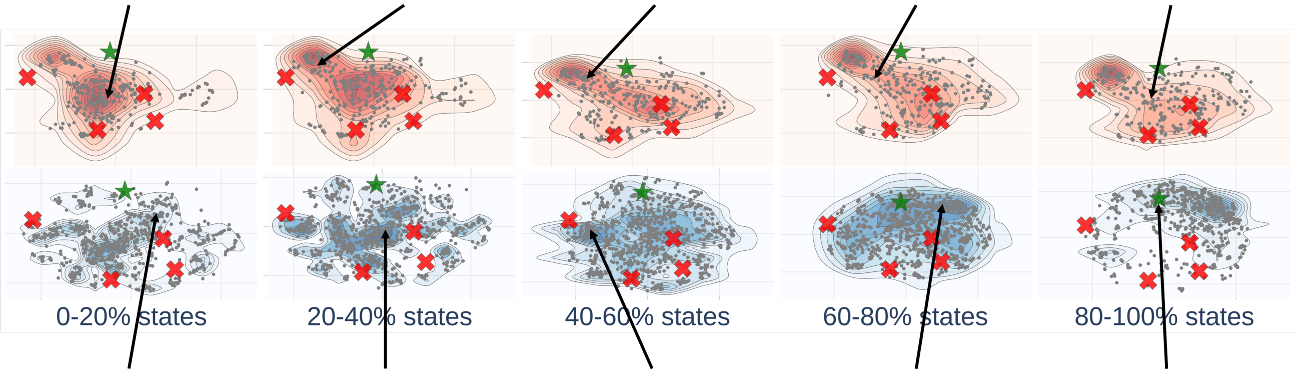
Let's break down the problem into steps to find the solution.

Now, rewrite the two equations in terms of 1, as follows: First equation is | = 66 - 2w and second is | = 48 - 2w.

The total cost is 50 + 32 = 82 rupees which is less than 100.

Therefore, the resultant solution is 37.25% tea and 62.75% milk.

Conclusion: The original price of the item was approximately $63.32. The answer is A



Step 1: Start by adding the positive numbers: adding 45 to -30 results in 15 since 15 > -15

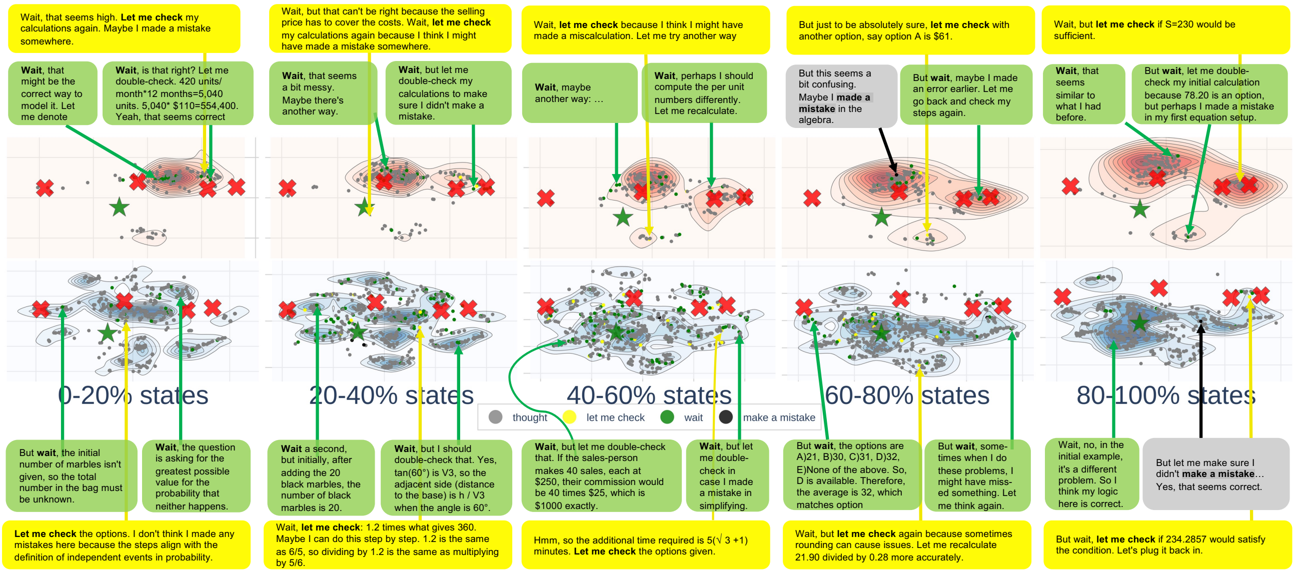
The distance traveled on the third side is 's' kilometers at a speed of 600 km/hr.

Therefore, to find the total sales, we need to divide the additional amount by 5% (which is 0.05).

Now, we multiply the common prime factors and the uncommon prime factors together to find the LCM.

Therefore, the answer is D)40

Figure 25: Case Study: Landscape of thoughts of Llama-3.1-70B on AQuA using CoT.



Wait, that seems high. **Let me check** my calculations again. Maybe I made a mistake somewhere.

Wait, that might be the correct way to model it. Let me denote

Wait, is that right? Let me double-check. 420 units/ month*12 months=5,040 units. 5,040* $110=554,400. Yeah, that seems correct

But **wait**, the initial number of marbles isn't given, so the total number in the bag must be unknown.

Wait, the question is asking for the greatest possible value for the probability that neither happens.

Let me check the options. I don't think I made any mistakes here because the steps align with the definition of independent events in probability.

Wait, but that can't be right because the selling price has to cover the costs. Wait, **let me check** my calculations again because I think I might have made a mistake somewhere.

Wait, that seems a bit messy. Maybe there's another way.

Wait, but let me double-check my calculations to make sure I didn't make a mistake.

Wait a second, but initially, after adding the 20 black marbles, the number of black marbles is 20.

Wait, but I should double-check that. Yes, tan(60°) is V3, so the adjacent side (distance to the base) is h / V3 when the angle is 60°.

Wait, **let me check**: 1.2 times what gives 360. Maybe I can do this step by step. 1.2 is the same as 6/5, so dividing by 1.2 is the same as multiplying by 5/6.

Wait, **let me check** because I think I might have made a miscalculation. Let me try another way

Wait, maybe another way: …

Wait, perhaps I should compute the per unit numbers differently. Let me recalculate.

Wait, but let me double-check that. If the sales-person makes 40 sales, each at $250, their commission would be 40 times $25, which is $1000 exactly.

Wait, but let me double-check in case I made a mistake in simplifying.

Hmm, so the additional time required is 5(√ 3 +1) minutes. **Let me check** the options given.

But just to be absolutely sure, **let me check** with another option, say option A is $61.

But this seems a bit confusing. Maybe I **made a mistake** in the algebra.

But **wait**, maybe I made an error earlier. Let me go back and check my steps again.

But **wait**, the options are A)21, B)30, C)31, D)32, E)None of the above. So, D is available. Therefore, the average is 32, which matches option

But **wait**, some-times when I do these problems, I might have miss-ed something. Let me think again.

Wait, but **let me check** again because sometimes rounding can cause issues. Let me recalculate 21.90 divided by 0.28 more accurately.

Wait, but **let me check** if S=230 would be sufficient.

Wait, that seems similar to what I had before.

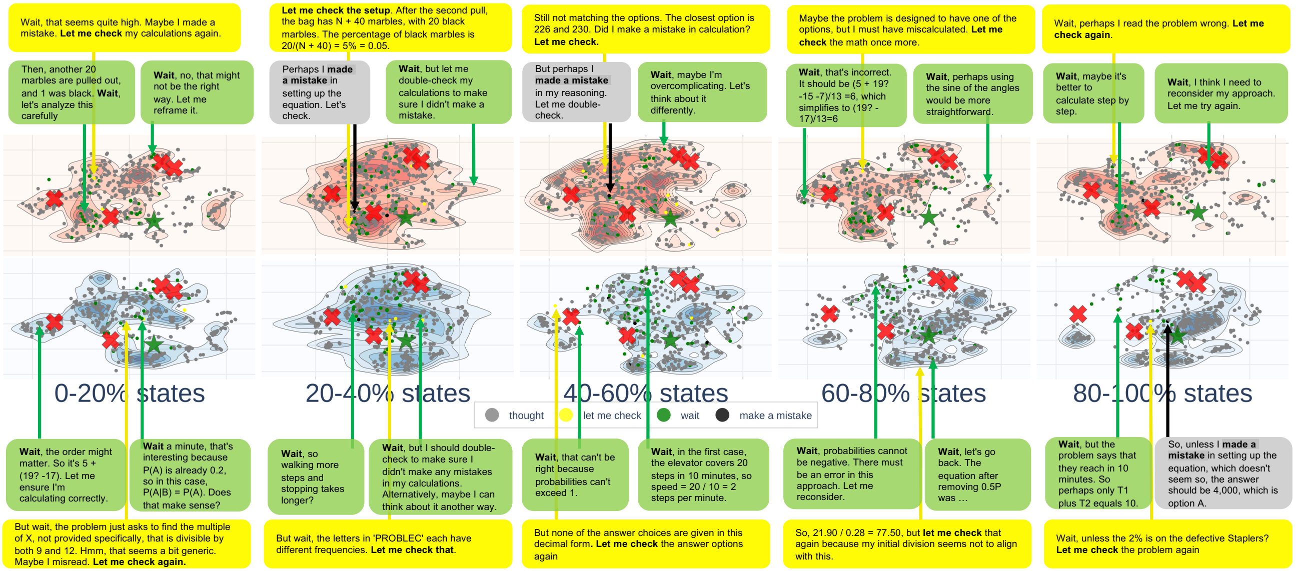
But **wait**, let me double-check my initial calculation because 78.20 is an option, but perhaps I made a mistake in my first equation setup.

Wait, no, in the initial example, it's a different problem. So I think my logic here is correct.

But let me make sure I didn't **make a mistake**… Yes, that seems correct.

But wait, **let me check** if 234.2857 would satisfy the condition. Let's plug it back in.

Figure 26: Landscape of DeepSeek-R1-Distill-Llama-70B using CoT on AQuA.



Wait, that seems quite high. Maybe I made a mistake. **Let me check** my calculations again.

Then, another 20 marbles are pulled out, and 1 was black. **Wait**, let's analyze this carefully

Wait, no, that might not be the right way. Let me reframe it.

Wait, the order might matter. So it's 5 + (19? -17). Let me ensure I'm calculating correctly.

Wait a minute, that's interesting because P(A) is already 0.2, so in this case, P(A|B) = P(A). Does that make sense?

But wait, the problem just asks to find the multiple of X, not provided specifically, that is divisible by both 9 and 12. Hmm, that seems a bit generic. Maybe I misread. **Let me check again.**

Let me check the setup. After the second pull, the bag has N + 40 marbles, with 20 black marbles. The percentage of black marbles is 20/(N + 40) = 5% = 0.05.

Perhaps I **made a mistake** in setting up the equation. Let's check.

Wait, but let me double-check my calculations to make sure I didn't make a mistake.

Wait, so walking more steps and stopping takes longer?

Wait, but I should double-check to make sure I didn't make any mistakes in my calculations. Alternatively, maybe I can think about it another way.

But wait, the letters in 'PROBLEC' each have different frequencies. **Let me check that**.

Still not matching the options. The closest option is 226 and 230. Did I make a mistake in calculation? **Let me check.**

But perhaps I **made a mistake** in my reasoning. Let me double-check.

Wait, maybe I'm overcomplicating. Let's think about it differently.

Wait, that can't be right because probabilities can't exceed 1.

Wait, in the first case, the elevator covers 20 steps in 10 minutes, so speed = 20 / 10 = 2 steps per minute.

But none of the answer choices are given in this decimal form**. Let me check** the answer options again

Maybe the problem is designed to have one of the options, but I must have miscalculated. **Let me check** the math once more.

Wait, that's incorrect. It should be (5 + 19? -15 -7)/13 =6, which simplifies to (19? - 17)/13=6

Wait, perhaps using the sine of the angles would be more straightforward.

Wait, probabilities cannot be negative. There must be an error in this approach. Let me reconsider.

Wait, let's go back. The equation after removing 0.5P was …

So, 21.90 / 0.28 = 77.50, but **let me check** that again because my initial division seems not to align with this.

Wait, perhaps I read the problem wrong. **Let me check again**.

Wait, maybe it's better to calculate step by step.

Wait, I think I need to reconsider my approach. Let me try again.

Wait, but the problem says that they reach in 10 minutes. So perhaps only T1 plus T2 equals 10.

So, unless I **made a mistake** in setting up the equation, which doesn't seem so, the answer should be 4,000, which is option A.

Wait, unless the 2% is on the defective Staplers? **Let me check** the problem again

Figure 27: Landscape of DeepSeek-R1-Distill-Qwen-1.5B using CoT on AQuA.