ARE LATENT REASONING MODELS EASILY INTERPRETABLE?

Connor Dilgren & Sarah Wiegreffe
Department of Computer Science
University of Maryland
College Park, MD, USA
{cdilgren,sarahwie}@umd.edu

ABSTRACT

Latent reasoning models (LRMs) have recently attracted significant research interest due to their lower inference cost compared to explicit chain-of-thought approaches. However, this efficiency comes at the cost of reduced interpretability: LRMs are more difficult for humans to monitor because they do not reason in natural language. This paper presents an initial investigation into LRM interpretability by examining two key questions on the Coconut and CODI models. First, we test the assumption made in prior work that latent reasoning tokens are necessary for model performance. We find that latent reasoning tokens are often unnecessary for LRMs' predictions; on logical reasoning datasets, LRMs can produce the same final answers without using latent reasoning tokens at all. This underutilization of reasoning tokens may partially explain why LRMs do not consistently outperform explicit reasoning methods, and raises doubts about the role of these tokens proposed in prior work. Second, when latent reasoning tokens *are* necessary, we investigate whether we can easily decode gold reasoning traces from them as a form of natural language explanation. Using a proposed backtracking method, we decode gold reasoning traces from latent reasoning tokens 71-93% of the time for correctly predicted instances when operands from the question are included, but only 24-36% of the time for incorrect predictions. This suggests that for correct predictions, LRMs are implementing the expected solution rather than an uninterpretable reasoning process. We find preliminary evidence that incorrect predictions can also be interpreted in this manner, though more robust methods are needed to reliably decode reasoning traces when models do not implement the gold reasoning trace.

1 INTRODUCTION

Reasoning methods such as chain-of-thought (CoT) (Wei et al., 2022) and tree-of-thought (Yao et al., 2023) have been shown to improve a Language Model (LM)'s performance in a range of tasks by having the LM solve problems in a step-by-step manner. This paradigm of additional inference-time token roll-out to improve performance is now ubiquitous, with most state-of-the-art model releases today being reasoning models (Ng, 2025). One can consider the generation of intermediate reasoning tokens as increasing the "width" of the network at inference-time, in a manner complementary to increasing its depth or dimensionality (i.e., LLM "scaling laws"; Kaplan et al. (2020)). Theoretical work has demonstrated that reasoning token generation increases the "effective depth" of the network by lengthening its longest pathways Feng et al. (2023); Li et al. (2023), and allows models to solve harder classes of problems (Merrill & Sabharwal, 2024; Nowak et al., 2024; Saunshi et al., 2025). Reasoning token generation has the added benefit of providing users with some insight into models' computational processes in natural language, which serve as a form of explanation. While the explicit reasoning chain is not always faithful to the model's true reasoning process (Wiegreffe et al., 2021; Turpin et al., 2023; Chen et al., 2025b), it has nonetheless been an important signal for users to calibrate their trust in a model's output (Baker et al., 2025).

However, the production of reasoning tokens at inference-time is fairly computationally intensive and not easily parallelizable, and many state-of-the art reasoning models (RMs) produce thousands

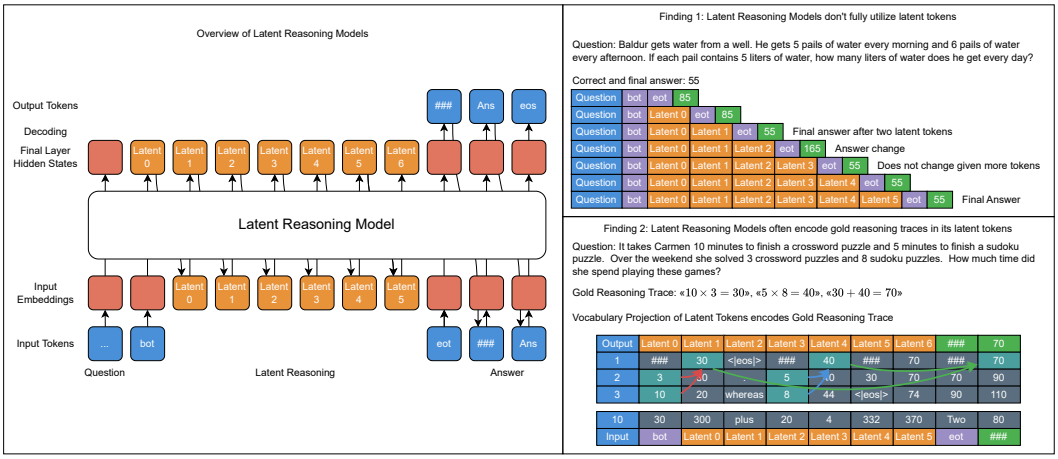


Figure 1: An overview of latent reasoning models (LRMs) and our main contributions. **Left:** LRM overview. Latent reasoning begins at the <bot> token. During this phase, the final layer hidden states are passed as the next input in the sequence. Standard autoregressive generation resumes after the <eot> token, and the model proceeds to answer the prompt. **Upper right:** LRMs tend to commit to a final answer well before exhausting its budget, indicating that they don't effectively use all available latent reasoning tokens. **Lower right:** Vocabulary projections of the latent tokens often encode gold reasoning traces, suggesting that the model follows an interpretable reasoning trace rather than an opaque one.

of tokens per query (Chen et al., 2025a; Yeo et al., 2025). An array of recent work has focused on improving RMs' inference-time efficiency (Qu et al., 2025; Zhu & Li, 2025; Liu et al., 2025; Sui et al., 2025; Feng et al., 2025; Alomrani et al., 2025), with proposed methods ranging from prompting- or decoding-based tricks (Wang et al., 2025), to fine-tuning models to encourage less reasoning token use (Luo et al., 2025), to dynamically allocating queries based on the necessity of reasoning (Singh et al., 2025). An approach that has shown promising recent results is that of latent reasoning models (LRMs; Figure 1), which proposes to make reasoning more efficient by forgoing the text decoding process altogether. Methods such as Deng et al. (2024a); Hao et al. (2025); Deng et al. (2025); Cheng & Durme (2024); Geiping et al. (2025) train models to autoregressively or recurrently generate additional intermediate latent "reasoning" states as needed. Latent reasoning architectures can also be motivated by the intuition that, in the purely computational view of Transformer architectures, the added constraint of decoding intermediate reasoning hidden states into text is an unneeded bottleneck on information flow; by not requiring reasoning states to be decoded into text, LRMs remove this bottleneck (Zhu et al., 2025b).

Unfortunately, unlike explicit reasoning models (ERMs), LRMs are less interpretable because they do not produce human-inspectable natural language reasoning tokens. This has led to increasing safety concerns about LRMs and calls to preserve explicit reasoning or "chain-of-thought monitorability" (Korbak et al., 2025). But do we have cause for concern with current LRMs? There is no current research on the extent to which LRMs emulate explicit reasoners. Prior work (Hao et al., 2025; Tan et al., 2025) has offered only limited case studies in support of claims of interpretability, with lack of standardized comparison across architectures or datasets. We set out to provide the first comprehensive study of latent reasoning interpretability. We answer two main research questions:

1. Are latent reasoning tokens necessary for model performance?
2. Are gold reasoning traces easily recoverable from latent reasoning tokens?

We first investigate whether latent reasoning tokens in current state-of-the-art LRMs are necessary to replicate models' performance, a prerequisite for meaningful interpretation. Somewhat counterintuitively (and at odds with the hypotheses put forth in prior work) we find that they are not: LRMs' predictions on logical reasoning datasets are often the same regardless of how many latent reasoning tokens the model is provided access to at inference time. This indicates either that current tasks are

too easy to truly test the benefits of latent reasoning tokens, or that the performance gains reported in prior work are from the model's training regimen and not additional test-time computation.

When models *do* require latent reasoning tokens, we next investigate whether we can find gold reasoning traces encoded in the latent reasoning tokens. We find that it is indeed possible to decode the gold reasoning traces using simple heuristics when models are correct, but less readily when models are incorrect. This suggests that, when making correct predictions, state-of-the-art LRMs follow the expected gold reasoning traces rather than implement some uninterpretable reasoning process. However, we are not *always* able to find the gold reasoning trace for some correctly predicted instances nor are we able to make sense of all incorrect traces, indicating the need for more robust interpretability methods that are also not reliant on an expected reasoning trace.

2 Related Work

2.1 Latent Reasoning Models

Latent reasoning models perform intermediate calculations in a continuous hidden state before answering. This is similar to explicit reasoning models (ERMs) that use chain-of-thought, except the intermediate states are not in natural language and thus aren't readily understandable by humans. The main paradigms of LRMs are activation-based methods, which iteratively process representations in a loop, and temporal hidden state-based methods, which allow models to integrate information over long sequences (Zhu et al., 2025b). In this paper, we study the width-based LRMs Coconut (Hao et al., 2025) and CODI (Shen et al., 2025). Width-based LRMs are a type of activation-based method where models pass hidden states (i.e., latent tokens) as the next token input (see Figure 1, left). We study these models both because they are increasingly common in the literature and because they are architecturally similar to ERMs. Coconut and CODI are two state-of-the-art models of this type that also have publicly-available source code.

During training, both the Coconut and CODI models learn to reason from ERMs. The Coconut model is instantiated as an ERM, and then, at each stage of a training curriculum, an explicit reasoning step is replaced with latent reasoning tokens until there are no explicit reasoning steps remaining. The CODI model trains an ERM alongside the LRM, and distills knowledge to the LRM by aligning the hidden states of a key token between the models.

During inference, Coconut and CODI operate in almost the same way (Figure 1, left). A special "beginning of thought" (<bot>) token is passed in to signal the start of latent reasoning. The model is then given a predetermined, dataset-specific number of latent reasoning tokens. Each input latent reasoning token is the final layer hidden state from the previous position, skipping the decoding phase of standard autoregressive language modeling. To signal the end of latent reasoning, a special "end of thought" (<eot>) token is passed as input, after which the model returns to standard autoregressive language modeling to produce a final answer in natural language. The only difference between Coconut and CODI during inference is that the final layer hidden state of CODI is additionally passed through a trained two-layer multi-layer perceptron before being input as the next token, allowing the model to better differentiate between latent and discrete tokens.

2.2 Interpreting Latent Reasoning Models

Limited work has been done on interpreting LRMs, with no existing dedicated work on this topic. Some works proposing latent reasoning architectures, including Hao et al. (2025); Shen et al. (2025), have included interpretability analysis, largely through case studies, but it is unclear to what extent these fully represent model behavior. Both works project latent hidden states to the vocabulary space using the unembedding matrix and observe whether intermediate numerical quantities for math questions appear in the top-k vocabulary tokens. Shen et al. (2025), for example, find preliminary evidence that the solutions to intermediate math problems in gold reasoning traces are present in the top-k tokens, and that the model also attends to relevant token positions in the input at those timesteps. Hao et al. (2025) inspect the probabilities assigned to various possible answer entities by latent hidden states (after vocabulary projection) to attempt to understand how answer choices are being considered or eliminated; they draw parallels to the heights of various entities in the test set's search trees. They also find that probability mass concentrates on fewer tokens as latent reasoning progresses. Tan et al. (2025) show for a specific instance that the embeddings of numbers

from the ground-truth reasoning chain have high cosine similarity with the model's latent hidden states. However, it is unclear to what extent all of these reported findings hold more generally across corpora, or whether they are predictive of models' correct vs. incorrect predictions.

From a theoretical angle, recent work (Zhu et al., 2025a; Gozeten et al., 2025) has demonstrated that LRMs with width-expansion can solve the directed graph reachability problem far more efficiently than ERMs, due to their ability to perform parallel breadth-first search. However, most current work on width-based latent reasoning has failed to outperform ERMs, indicating that the gap between theory and practice remains high.

Though a substantially different architecture from the models we consider, some work has interpreted the hidden states of recurrent *depth* (as opposed to dynamic width via token rollouts) LRMs. Geiping et al. (2025) perform PCA on the hidden representations of their recurrent-depth architecture, and demonstrate that representations' trajectories during recursion follow distinct geometric patterns. Lu et al. (2025) adapt vocabulary projection methods to show evidence against both iterative refinement and structured CoT-like reasoning in the same model, finding that projection to the vocabulary space is less meaningful when depth is increased.

3 Experimental Details

This section details the datasets, LRMs, and base LLMs used in our experiments in §4 and §5.

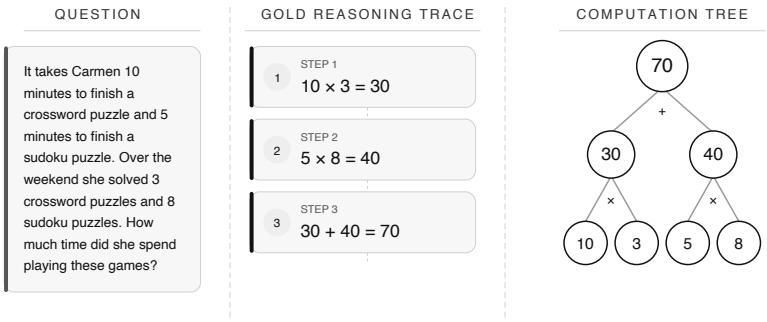


Figure 2: The question, gold reasoning trace, and derived computation tree for sample 199 of GSM8k-Aug's filtered test split.

Datasets. We perform experiments on three datasets commonly studied in prior work on LRMs: Grade School Math 8k-Augmented (GSM8k-Aug; Deng et al., 2024b), Proof and Ontology-Generated Question-Answering (PrOntoQA; Saparov & He, 2023), and Proof with Search Question-Answering (ProsQA; Hao et al., 2025). Figure 7 contains examples from each dataset, and Table 2 contains dataset statistics.

GSM8k-Aug is a dataset of basic arithmetic word problems, each containing a gold label final answer and a gold step-by-step reasoning trace (with 1-8 steps) leading to the final answer. Figure 2 contains an example from this dataset. GSM8k-Aug is a superset of GSM8k Cobbe et al. (2021), augmented with additional training instances generated by GPT-4 OpenAI et al. (2024); both datasets have the same test set. We apply two filtering steps to the test split: first, we remove samples with missing gold reasoning traces. Second, we remove samples where the result of the last step in the gold reasoning trace does not match the correct answer.[1] After filtering, the test split contains 1,194 samples out of the original 1,319 test samples.

The math problems in GSM8k-Aug can be solved through multiple valid gold reasoning traces. Because our proposed experiments will check similarity to the gold reasoning trace, we augment the original gold reasoning traces with additional valid reasoning traces from the MultiChain GSM8k-Aug dataset (Deng et al., 2025). After augmentation, the number of gold reasoning traces per sample ranges from 1 to 10, with a median of 5.

[1]These occur when some or all of the gold reasoning trace in GSM8k is explained in natural language only and does not have a corresponding annotated calculation.

PrOntoQA and ProsQA are both logical reasoning datasets that require 6 and 3–6 deductive reasoning steps, respectively, to solve. The task scenario for each dataset is to determine whether an entity belongs to a stated category, given a set of hierarchical "is-a" relationships. PrOntoQA frames the question as true or false, while ProsQA asks which category an entity belongs to given two choices; see Figure 7. The given relationships can be modeled as directed acyclic graphs, where there is at least one path from the entity node to the stated category. ProsQA generally has more distractor paths and thus requires more complex planning or search ability than PrOntoQA; it was proposed by Hao et al. (2025) to resolve the shortcomings of PrOntoQA for testing search in LRMs. The test splits of PrOntoQA and ProsQA have 800 and 500 samples, respectively.

Table 1: Performance of the explicit reasoning, non-reasoning, Coconut, and CODI models, all instantiated with GPT-2 Small, on GSM8k-Aug, PrOntoQA, and ProsQA. Published results for the explicit reasoning, non-reasoning, and Coconut models come from Hao et al. (2025). Published results for the CODI model is from Shen et al. (2025). For fair comparison with prior work, GSM8k-Aug results are computed on the full 1,319 test instances. *: this model checkpoint was released by the authors; we did not re-train.

Method	**GSM8k-Aug**		**PrOntoQA**		**ProsQA**	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
Explicit Reasoning Model (our replication)	41.6	31.0	99.3	92.7	74.2	51.6
Explicit Reasoning Model (published)	42.9	25.0	98.8	92.5	77.5	49.4
Non-reasoning Model (our replication)	16.8	3.2	87.9	3.0	76.0	9.5
Non-reasoning Model (published)	16.5	2.2	93.8	3.0	76.7	8.2
Coconut (our replication)	33.1	9.2	99.0	9.0	98.0	15.5
Coconut (published)	34.1	8.2	99.8	9.0	97.0	14.2
CODI (our replication)	42.2*	12.3	95.1	12.0	81.6	18.2
CODI (published)	43.7	-	-	-	-	-

Models. Following prior work, we use GPT-2 Small (Radford et al., 2019) as our base pretrained checkpoint and fine-tune it for each dataset using the various latent training regimens proposed in prior work– specifically, Coconut and CODI; see §2.1 for details. We additionally fine-tune two baselines: an ERM (i.e., a model that uses chain-of-thought reasoning in natural language), and a non-reasoning model (i.e., a model that immediately answers without reasoning). Following prior work, we fine-tune each of the four model types separately for each dataset (GSM8k-Aug, ProsQA, or PrOntoQA) using the provided training code of Hao et al. (2025); Shen et al. (2025), resulting in twelve total models.[2] Following Hao et al. (2025) and Shen et al. (2025), we train and evaluate our Coconut and CODI models use 6 latent reasoning tokens for all samples. Performance of our replications is in Table 1. The ERM outperforms Coconut on GSM8k-Aug by 8.5 percentage points, while CODI slightly exceeds the ERM by 0.6 percentage points. Coconut outperforms both CODI and the ERM on ProsQA, by 16.4 and 23.8 percentage points, respectively. All reasoning models achieve greater than 95% accuracy on PrOntoQA.

4 Are latent reasoning tokens necessary for model performance?

We first investigate whether trained LRMs effectively *use* their additional computational power (i.e., latent reasoning tokens) at inference time by testing their ability to maintain their predictions under forced early termination of reasoning. If a model consistently predicts the same final answer with no or fewer latent reasoning tokens than the standard six, then either 1) the task is too easy to truly test the benefits of the latent reasoning architecture, and/or 2) performance gains of the latent reasoning architecture are coming from the training regimen and *not* from additional token roll-out at inference-time.

[2]Except for GPT2-Small CODI on GSM8k-Aug, for which we use the provided checkpoint: `https://huggingface.co/zen-E/CODI-gpt2`.

4.1 METHOD

To determine the effective number of reasoning tokens required by a model to arrive at its final answer, we control the number of latent reasoning tokens used by prematurely inserting the "end-of-thought" token (see Figure 1, upper right). Doing so terminates latent reasoning and prompts the model to produce a final answer in natural language ("forced early stopping"). If l is the number of reasoning tokens, we compare final answer predictions after the full $l = 6$ tokens to the model's answer given a reduced number of reasoning tokens, $l \in [0, 1, 2, 3, 4, 5]$, using the following metrics:

1. **First match**: the minimum number of reasoning tokens at which the model's answer matches its answer given the full set of reasoning tokens. In Figure 1 (upper right), the first match occurs at $l = 2$.
2. **Stable match**: the minimum number of reasoning tokens at which the model's answer matches its answer given the full set of reasoning tokens *and* remains unchanged given additional reasoning tokens. In Figure 1 (upper right), the stable match occurs at $l = 4$.

We also run this analysis on the ERMs as a baseline. Since latent reasoning tokens are trained to replace full reasoning steps (§2.1), we evaluate the ERMs by removing complete steps.

4.2 RESULTS

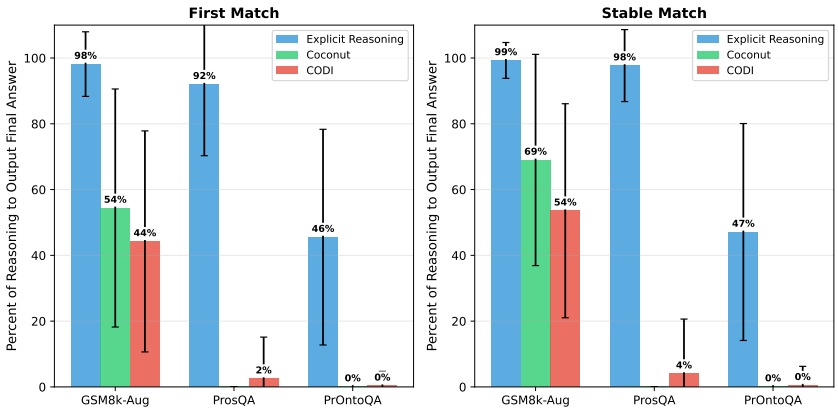


Figure 3: Early stopping experiment results. The y-axis indicates the percent of the model's reasoning chain needed for the model to output the same answer given the full reasoning chain. For the LRMs (Coconut and CODI), the reasoning chain is measured by the number of latent reasoning tokens. For the ERMs, the reasoning chain is measured by the number of reasoning steps.

Results for all datasets are presented in Figure 3. We make a surprising finding: unlike for GSM8k-Aug, where all models require at least 44% of the latent reasoning tokens (or steps, in the explicit reasoning case), Coconut and CODI models rarely need *any* of their latent reasoning tokens to make stable predictions on PrOntoQA or ProsQA, and can generally converge to their final answer without any reasoning steps at all. The explicit CoT model, by comparison, still requires 46% and 92% of its reasoning tokens for PrOntoQA and ProsQA, respectively.

This result contradicts the analysis in Hao et al. (2025), which concludes that the Coconut model uses a parallelized breadth-first search to solve problems in PrOntoQA and ProsQA. It is possible that the Coconut model still performs some form of search when latent reasoning is not limited as it is in our experiments, but our result demonstrates that search in latent token space at inference time is ***not necessary*** for Coconut to achieve its strong performance. In other words, Coconut is capable of answering ProsQA and PrOntoQA questions either by performing search during standard inference (i.e., without explicit or implicit reasoning steps) or by not using search at all. Coconut's strong performance on the ProsQA dataset may instead be due to supervision on ProsQA examples during training, as opposed to the use of latent reasoning tokens at inference time. Future studies should first verify that latent tokens are necessary for the chosen dataset before analyzing how the latent tokens are used.

In contrast, on the GSM8k-Aug dataset, we observe that LRMs do use their reasoning tokens, though still at lower rates than the explicit model. The LRMs generally converge to their final answer given half to two-thirds of the full reasoning chain, while the ERM generally uses the full reasoning chain before outputting its final answer. This also reveals an inefficiency of LRMs: unlike ERMs, the number of reasoning tokens in Coconut and CODI is fixed rather than being determined dynamically by the model per instance.

The overall underutilization of latent reasoning tokens that we observe across all three datasets may partially explain why LRMs do not consistently surpass ERM performance (Table 1). Coconut has an 8.5% performance gap relative to the ERM on GSM8k-Aug, while CODI matches the ERM (Table 1). The models' tendency to converge prematurely suggests that they fail to exploit the additional computational bandwidth of their later latent reasoning tokens. Future work could address this from two angles: improving performance by training models to better utilize their full reasoning budget, or improving efficiency by introducing early stopping mechanisms that terminate reasoning once the LRM has reached a stable prediction.

5 Are gold reasoning traces easily recoverable from latent reasoning states?

When latent reasoning tokens *are* necessary for model performance, we next ask: can we easily decode gold reasoning traces from them? While prior work has projected latent tokens back to the vocabulary space for interpretation (§2.2), this has been done either on only a few case-studies (Hao et al., 2025; Shen et al., 2025) or in search of intermediate answer quantities rather than the full trace (Shen et al., 2025), and only on correct predictions.

5.1 Gold Reasoning Trace Backtracking Experiment

We use the popular vocabulary projection technique (or "logit lens"; nostalgebraist (2020); Geva et al. (2021)) to map latent tokens back to the model's vocabulary space. This is done by multiplying the residual stream after the final layer (and final LayerNorm) with the model's unembedding matrix to obtain an (unnormalized) distribution over the vocabulary. We repeat this at each latent token position, obtaining the top-10 natural language tokens (i.e., rows of the unembedding matrix) with the highest dot product against each latent token[3]; this is equivalent to how a natural language token would be decoded should the model have been operating as an ERM. Prior work has qualitatively observed the presence of certain intermediate computations in the top-k tokens; we automate this process. If the intermediate steps from the gold reasoning trace appear prominently in the vocabulary projections of the latent reasoning tokens, this suggests that the model is following these steps internally, even though it never explicitly outputs them.

We use a backtracking search to check whether a complete gold reasoning trace is present in the top-10 vocabulary projections of the latent tokens. To facilitate this search, we first represent the gold reasoning traces as computation trees. As shown in Figure 2, we set the final answer as the root node, and the operands in the final step as its child nodes. If any of the final step operands are an intermediate result from a previous step, then we add the operands from that previous step as its children. We continue until all steps have been added to the computation tree. We only analyze GSM8k-Aug in this experiment, since §4 demonstrated that LRMs do not utilize their reasoning tokens on PrOntoQA and ProsQA.

Next, we extract the top-10 tokens from all latent reasoning tokens using vocabulary projection. The backtracking search then proceeds as follows: starting at the correct answer (the root node), we check whether its operands (child nodes) appear in the top-k tokens at any previous position. For each child node found, we recursively check whether its children appear at even earlier positions. A child node must appear at an earlier position than its parent node. The ground truth tree is considered "found" if all nodes in the tree are successfully located in this manner. This search is conducted both with and without allowing question tokens to be operands. When searching for a computation tree with question tokens, numbers given in the question may be used as operands in the computation tree. Figure 4 shows a successfully found computation tree for the Coconut model.

[3]The top-10 tokens capture at least 90% of the probability mass over the vocabulary for the median GSM8k-Aug validation sample for both Coconut and CODI.

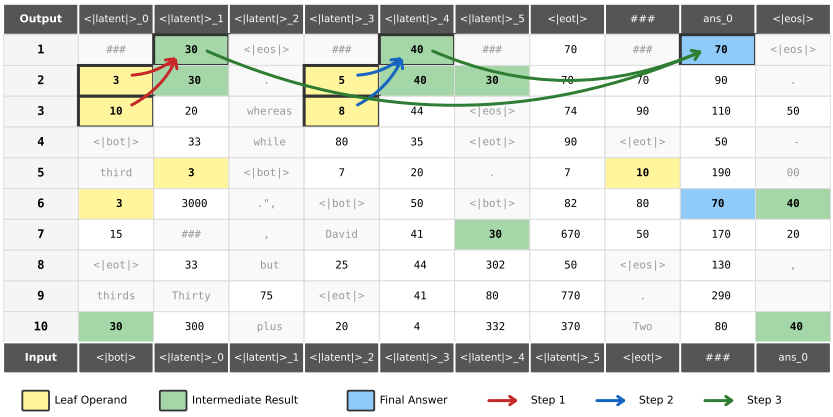


Figure 4: Found gold label computation tree in Coconut's vocabulary projections, from sample 199 of GSM8k-Aug's filtered test split. The model answered this question correctly and the top vocabulary projections represent the gold reasoning trace in Figure 2.

We run this analysis for correctly and incorrectly predicted samples. For correctly predicted samples, the backtracking search starts at the top-1 token at the answer position, which is the final answer in the gold reasoning traces. For incorrectly predicted samples, we look for the correct answer in the top-10 vocabulary projections at the answer position. If the correct answer is present, then the search for the gold reasoning traces proceeds normally. Table 3 shows that 46.5% and 49.9% of incorrectly predicted samples for Coconut and CODI, respectively, have the correct answer in the top-10 vocabulary projection at the answer position.

To verify that the latent reasoning tokens do not represent arbitrary reasoning traces, for each sample, we randomly select n reasoning traces from other GSM8k-Aug problems with the same number of steps. Then, we check whether any of these reasoning traces can also be found using the backtracking search method. If the top-k threshold used in the vocabulary projection is too high, then these random reasoning traces should be found at rates comparable to the gold reasoning traces. We use $n = 1$ and $n = 5$.

An inherent limitation of vocabulary projection is that it can only observe single-token concepts. To account for this, we assume that the first non-zero integer token of a multi-token number represents the full number. E.g., we assume the decimal "0.5" is represented by "5".

Another limitation of our backtracking method is that it requires known gold reasoning traces, and so cannot be used to discover other reasoning traces that the model may use. We use this backtracking method to gather evidence on whether LRMs implement expected reasoning strategies in latent space. Access to gold traces is necessary because we are testing alignment with known solutions.

5.2 Results

Figure 5 shows that the LRMs generally do encode the gold label reasoning trace for correctly answered samples. The Coconut model encodes the original gold reasoning trace in 54% of correctly answered samples. This increases to 65% when including additional valid reasoning traces from the MultiChain dataset, and then to 93% when also including numbers from the question. The CODI model encodes the gold reasoning traces at a much lower rate for correctly answered samples: only 5% encode the original gold reasoning trace. This increases to only 8% when considering the MultiChain dataset. This is because the CODI model generally does not encode the numbers from the question into its latent reasoning tokens, at least not in a way that is detectable with vocabulary projection. When we do include the numbers from the question, the CODI model's encoding of the original gold reasoning trace and any gold reasoning trace jumps to 61% and 71%, respectively. Figure 9 shows an example of the CODI model encoding the intermediate results but not the values already given in the question.

For correctly answered problems, the gold reasoning traces for a given sample are substantially more represented than randomly selected reasoning traces from other samples in the dataset. The best of five random reasoning traces are represented only 7% and 2% of the time for Coconut and CODI, respectively, even when including question tokens. This baseline result gives us confidence that the top-10 vocabulary projections are not expressive enough to represent arbitrary reasoning traces.

Somewhat surprisingly, the LRMs sometimes represent the gold reasoning traces even for incorrectly answered problems. The Coconut and CODI models represent at least one of the gold reasoning traces 36% and 24% of the time, respectively, when including numbers in the question. In these cases, the model encodes an incorrect reasoning trace more strongly than the gold reasoning trace. Figure 11 shows an example where the correct steps are encoded, but not as strongly as an incorrect set of steps.

There is a drop-off in gold reasoning trace representation as the reasoning trace length increases beyond 3 steps, as shown in Figure 6. Including question tokens, Coconut declines from 99% at two steps to 38% at five steps. CODI declines from 85% at two steps to just 20% at five steps. This degradation reflects a limitation in the models' capacity to maintain longer reasoning chains.

The results of this experiment provide preliminary evidence that LRMs solve elementary math problems in the same way that ERMs do: by calculating intermediate steps and composing them to output a final answer. The main evidence for this is that the gold label reasoning traces are consistently present when the model is correct compared to when the model is incorrect, and this is not explained simply by overly expressive vocabulary projections. We interpret this to mean that LRMs are learning to compress but still encode gold reasoning traces, rather than abandoning them for less understandable ways of solving elementary math problems.

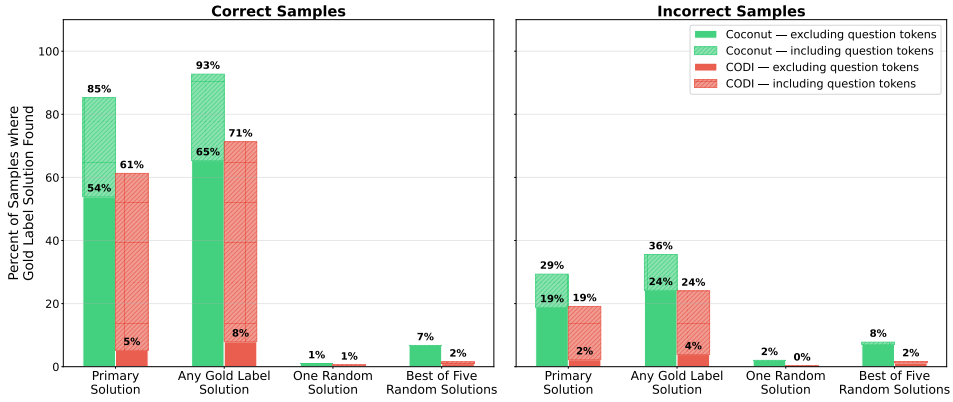


Figure 5: Percent of gold reasoning traces found in the vocabulary projections of latent tokens for correctly and incorrectly answered problems. The "Primary Solutions" are solutions from the original GSM8k-Aug dataset. The "Any Gold Label Solution" category includes primary solutions and additional solutions from the MultiChain GSM8k-Aug dataset (Deng et al., 2025).

5.3 Error Analysis

When the backtracking search fails to find an encoded gold reasoning trace, how is the Coconut model solving the problem? We find evidence against the worst case scenario, where the LRM arrives at the correct answer in a completely uninterpretable way. Instead, we find three main reasons why the backtracking search can fail even when the model gets the correct answer: a valid reasoning trace may be missing from the set of known reasoning traces, vocabulary projection does not encode multi-token concepts an easily identifiable way, or most of but not the entire gold reasoning trace may be encoded.

Figure 13 shows an example where the model is following a valid reasoning trace that is not in the set of known reasoning traces. Specifically, the model skips Step 2 in the gold reasoning trace, which calculates $36 + 40 = 76$. Instead of calculating and storing this intermediate result, the Coconut model changes the last step from $76 + 46 = 122$ to an equivalent $36 + 40 + 46 = 122$. The MultiChain GSM8k-Aug dataset did not contain this alternative reasoning trace because its augmentations

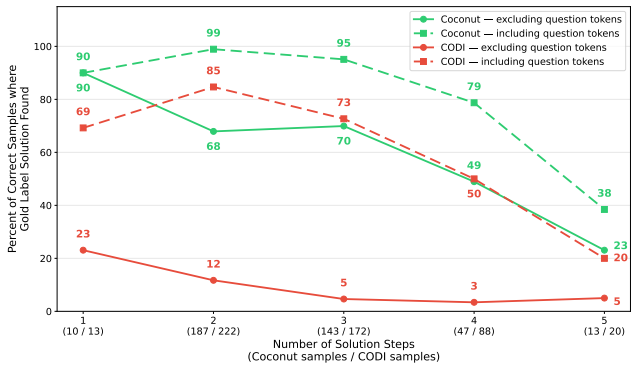


Figure 6: Percent of any gold reasoning trace found in the vocabulary projections of latent tokens for correctly answered problems by reasoning trace length. Reasoning traces with more than 5 steps are not shown due to low sample counts (< 3).

preserve the number of steps used in the original reasoning trace, and this modification reduces the step count from 4 to 3. To avoid this error, we'd need to enumerate all valid ways of solving the problem, which is impractical and often impossible.

Figure 15 shows an example where vocabulary projection limits our ability to identify decimals, percentages, and multi-token numbers generally that are encoded in a latent thought. The first step of the gold reasoning trace calculates 30% of 120. The gold reasoning trace represents the 30% as 30/100, which is equivalent, but the model does not represent the 100 in its vocabulary projection. Instead, it's likely that the "30" token in the top-2 of the vocabulary projection of the first latent token represents this percentage. But since the model and the gold reasoning trace represent this percentage differently, the backtracking search fails.

Figure 17 shows an example where Coconut's vocabulary projections contain a partial gold reasoning trace. In this sample, Step 3, $6 + 15 = 21$, is missing, so the intermediate result 21 is not encoded. However, the model is still able to calculate the result of the next and final step, 84. There are at at least two possibilities for this. It may encode the final step as $(6 + 15) * 4 = 84$, which would make it more like a previously unknown valid reasoning trace. Or, it may be encoding 21 at the final reasoning position, and vocabulary projection incorrectly extracts it as 2100 and 210, which are shown in the table.

This error analysis suggests that our backtracking search results provide a lower bound on LRM interpretability, with failures stemming from methodological limitations rather than fundamentally uninterpretable model behavior. More robust reasoning trace finding techniques that handle equivalent reformulations and flexible numerical encodings would likely recover a higher proportion of encoded reasoning traces.

6 Conclusion

This paper investigated LRM interpretability, which is essential for deployment where monitorability is required. Our findings reveal two key insights. First, LRMs do not fully utilize their latent reasoning tokens. On GSM8k-Aug, LRMs generally utilize half to two-thirds of their latent reasoning tokens, and on logical reasoning datasets, LRMs determine their final answer without latent reasoning at all. This suggests that either current datasets are insufficiently challenging or that performance gains in prior work stem from the training regimen rather than additional inference-time computation. Second, when reasoning tokens are used, gold reasoning traces can be recovered from correct predictions using simple heuristics, suggesting that LRMs implement expected reasoning traces rather than opaque reasoning processes. However, our recovery method fails more often for incorrect predictions, and even some correctly predicted instances are missing part or all of the gold reasoning trace. We suggest future work carefully select datasets that require latent reasoning and develop more robust interpretability methods that do not rely on knowing expected reasoning traces.

REFERENCES

Mohammad Ali Alomrani, Yingxue Zhang, Derek Li, Qianyi Sun, Soumyasundar Pal, Zhanguang Zhang, Yaochen Hu, Rohan Deepak Ajwani, Antonios Valkanas, Raika Karimi, Peng Cheng, Yunzhou Wang, Pengyi Liao, Hanrui Huang, Bin Wang, Jianye Hao, and Mark Coates. Reasoning on a Budget: A Survey of Adaptive and Controllable Test-Time Compute in LLMs, July 2025. URL http://arxiv.org/abs/2507.02076. arXiv:2507.02076 [cs].

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y. Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation, 2025. URL https://arxiv.org/abs/2503.11926.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards Reasoning Era: A Survey of Long Chain-of-Thought for Reasoning Large Language Models, July 2025a. URL http://arxiv.org/abs/2503.09567. arXiv:2503.09567 [cs].

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. Reasoning models don't always say what they think, 2025b. URL https://arxiv.org/abs/2505.05410.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations, 2024. URL https://arxiv.org/abs/2412.13171.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Jingcheng Deng, Liang Pang, Zihao Wei, Shichen Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Latent reasoning in llms as a vocabulary-space superposition, 2025. URL https://arxiv.org/abs/2510.15522.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step, 2024a. URL https://arxiv.org/abs/2405.14838.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation, 2024b. URL https://openreview.net/forum?id=9cumTvvlHG.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards Revealing the Mystery behind Chain of Thought: A Theoretical Perspective. November 2023. URL https://openreview.net/forum?id=qHrADgAdYu&noteId=JgRIVMxGoT.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. Efficient Reasoning Models: A Survey. *Transactions on Machine Learning Research*, May 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=sySqlxj8EB.

Jonas Geiping, Sean Michael McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=S3GhJooWIC.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446.

Halil Alperen Gozeten, M. Emrullah Ildiz, Xuechen Zhang, Hrayr Harutyunyan, Ankit Singh Rawat, and Samet Oymak. Continuous chain of thought enables parallel exploration and reasoning, 2025. URL https://arxiv.org/abs/2505.23648.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio, Joe Benton, Joseph Bloom, Mark Chen, Alan Cooney, Allan Dafoe, Anca Dragan, Scott Emmons, Owain Evans, David Farhi, Ryan Greenblatt, Dan Hendrycks, Marius Hobbhahn, Evan Hubinger, Geoffrey Irving, Erik Jenner, Daniel Kokotajlo, Victoria Krakovna, Shane Legg, David Lindner, David Luan, Aleksander Mądry, Julian Michael, Neel Nanda, Dave Orr, Jakub Pachocki, Ethan Perez, Mary Phuong, Fabien Roger, Joshua Saxe, Buck Shlegeris, Martín Soto, Eric Steinberger, Jasmine Wang, Wojciech Zaremba, Bowen Baker, Rohin Shah, and Vlad Mikulik. Chain of thought monitorability: A new and fragile opportunity for ai safety, 2025. URL https://arxiv.org/abs/2507.11473.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of Thought Empowers Transformers to Solve Inherently Serial Problems. October 2023. URL https://openreview.net/forum?id=3EWTEy9MTM.

Yue Liu, Jiaying Wu, Yufei He, Ruihan Gong, Jun Xia, Liang Li, Hongcheng Gao, Hongyu Chen, Baolong Bi, Jiaheng Zhang, Zhiqi Huang, Bryan Hooi, Stan Z. Li, and Keqin Li. Efficient Inference for Large Reasoning Models: A Survey, August 2025. URL http://arxiv.org/abs/2503.23077. arXiv:2503.23077 [cs].

Wenquan Lu, Yuechuan Yang, Kyle Lee, Yanshu Li, and Enqi Liu. Latent chain-of-thought? decoding the depth-recurrent transformer. In *The First Workshop on the Application of LLM Explainability to Reasoning and Planning, COLM*, 2025. URL https://arxiv.org/abs/2507.02199.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning, 2025. URL https://arxiv.org/abs/2501.12570.

William Merrill and Ashish Sabharwal. The Expressive Power of Transformers with Chain of Thought, April 2024. URL http://arxiv.org/abs/2310.07923. arXiv:2310.07923 [cs].

Andrew Ng. Top stories of 2025. The Batch, DeepLearning.AI, 2025. URL https://www.deeplearning.ai/the-batch/issue-333/.

nostalgebraist. Interpreting gpt: the logit lens. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens, 2020. LessWrong blog post.

Franz Nowak, Anej Svete, Alexandra Butoi, and Ryan Cotterell. On the Representational Capacity of Neural Language Models with Chain-of-Thought Reasoning. In *ACL 2024*. arXiv, June 2024. doi: 10.48550/arXiv.2406.14197. URL http://arxiv.org/abs/2406.14197. arXiv:2406.14197 [cs].

OpenAI, 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, 2024. URL https://arxiv.org/abs/2303.08774.

Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, Peng Li, Wei Wei, Jing Shao, Chaochao Lu, Yue Zhang, Xian-Sheng Hua, Bowen Zhou, and Yu Cheng. A Survey of Efficient Reasoning for Large Reasoning Models: Language, Multimodality, and Beyond, March 2025. URL http://arxiv.org/abs/2503.21614. arXiv:2503.21614 [cs].

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=qFVVBzXxR2V.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=din0lGfZFd.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.36. URL https://aclanthology.org/2025.emnlp-main.36/.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card, 2025. URL https://arxiv.org/abs/2601.03267.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models. *Transactions on Machine Learning Research*, April 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=HvoG8SxggZ.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. Think silently, think fast: Dynamic latent compression of LLM reasoning chains. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=AQsko3PPUe.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting, 2023. URL https://arxiv.org/abs/2305.04388.

Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi Zhou. Wait, We Don't Need to "Wait"! Removing Thinking Tokens Improves Reasoning Efficiency, June 2025. URL http://arxiv.org/abs/2506.08343. arXiv:2506.08343 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022. URL https://arxiv.org/abs/2201.11903.

Sarah Wiegreffe, Ana Marasović, and Noah A. Smith. Measuring association between labels and free-text rationales. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10266–10284, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.804. URL https://aclanthology.org/2021.emnlp-main.804/.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL https://arxiv.org/abs/2305.10601.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying Long Chain-of-Thought Reasoning in LLMs, February 2025. URL http://arxiv.org/abs/2502.03373. arXiv:2502.03373 [cs].

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=UdOEZgWJLc.

Jason Zhu and Hongyu Li. Towards Concise and Adaptive Thinking in Large Reasoning Models: A Survey, July 2025. URL http://arxiv.org/abs/2507.09662. arXiv:2507.09662 [cs].

Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, Tianle Cai, Taylor Kergan, Assel Kembay, Andrew Smith, Chenghua Lin, Binh Nguyen, Yuqi Pan, Yuhong Chou, Zefan Cai, Zhenhe Wu, Yongchi Zhao, Tianyu Liu, Jian Yang, Wangchunshu Zhou, Chujie Zheng, Chongxuan Li, Yuyin Zhou, Zhoujun Li, Zhaoxiang Zhang, Jiaheng Liu, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A survey on latent reasoning, 2025b. URL https://arxiv.org/abs/2507.06203.

A Appendix

GSM8k-Aug

Question: "Out of 600 employees in a company, 30% got promoted while 10% received bonus. How many employees did not get either a promotion or a bonus?"
Steps: ["«600 * 30/100 = 180»", "«600 * 10/100 = 60»", "«180 + 60 = 240»", "«600 − 240 = 360»"]
Answer: "360"

PrOntoQA

Question: "Numpuses are not wooden. Vumpuses are lempuses. Rompuses are not dull. Each lorpus is a wumpus. Every gorpus is moderate. Each vumpus is not discordant. Zumpuses are not spicy. Shumpuses are windy. Brimpuses are grimpuses. Each grimpus is a rompus. Brimpuses are zumpuses. Each impus is not opaque. Lorpuses are not mean. Brimpuses are large. Grimpuses are shumpuses. Numpuses are impuses. Shumpuses are numpuses. Lempuses are hot. Numpuses are sterpuses. Shumpuses are gorpuses. Each yumpus is wooden. Every grimpus is orange. Each vumpus is a brimpus. Max is a vumpus. Max is a lorpus. True or false: Max is not wooden."
Steps: ["Max is a vumpus. Each vumpus is a brimpus.", "Max is a brimpus. Brimpuses are grimpuses.", "Max is a grimpus. Grimpuses are shumpuses.", "Max is a shumpus. Shumpuses are numpuses.", "Max is a numpus. Numpuses are not wooden.", "Max is not wooden."]
Answer: "True"

ProsQA

Question = "Every kerpus is a yumpus. Every bompus is a boompus. Every vumpus is a felpus. Sally is a vumpus. Every yimpus is a jompus. Every yerpus is a jelpus. Every kerpus is a terpus. Every bompus is a wumpus. Every rempus is a terpus. Every yerpus is a yimpus. Every rempus is a kerpus. Every wumpus is a kerpus. Every impus is a kerpus. Tom is a bompus. Every bompus is a timpus. Sally is a yerpus. Every yumpus is a terpus. Every yumpus is a zhorpus. Every bompus is a impus. Every wumpus is a zhorpus. Every yerpus is a jompus. Every yimpus is a vumpus. Every zumpus is a yumpus. Every zumpus is a rempus. Every zumpus is a storpus. Every timpus is a yumpus. Every impus is a timpus. Every timpus is a rempus. Tom is a impus. Every bompus is a zumpus. Tom is a lempus. Sally is a jompus. Every jelpus is a yimpus. Every rempus is a wumpus. Tom is a rempus. Every yerpus is a vumpus. Every jelpus is a jompus. Every impus is a rempus. Every jelpus is a vumpus. Sally is a storpus. Is Tom a jompus or zhorpus?"
Steps = ["Tom is a bompus.", "Every bompus is a wumpus.", "Every wumpus is a zhorpus."]
Answer = "Tom is a zhorpus."

Figure 7: Example instances from each dataset.

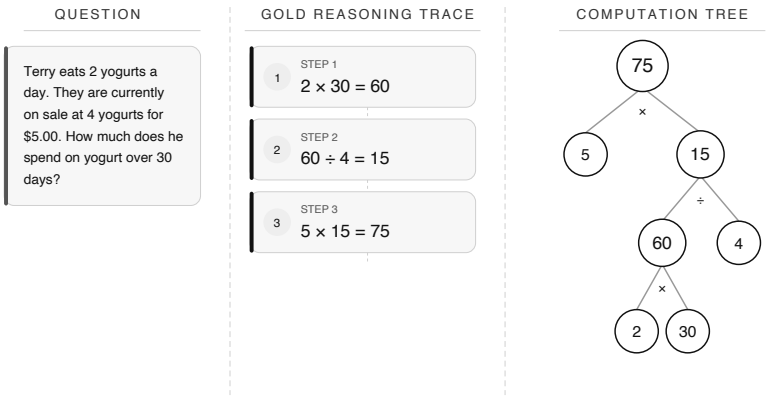
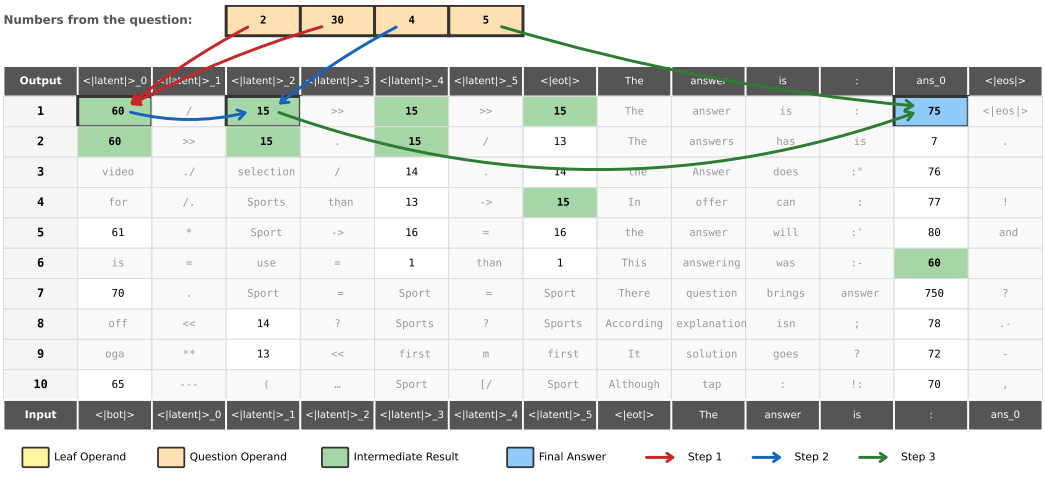


Figure 8: The question, gold reasoning trace, and derived computation tree for sample 31 of GSM8k-Aug's filtered test split.



Output	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	The	answer	is	:	ans_0	<|eos|>
1	60	/	15	>>	15	>>	15	The	answer	is	:	75	<|eos|>
2	60	>>	15	.	15	/	13	The	answers	has	is	7	.
3	video	./	selection	/	14	.	14	the	Answer	does	:"	76	
4	for	/.	Sports	than	13	->	15	In	offer	can	:	77	!
5	61	*	Sport	->	16	=	16	the	answer	will	:'	80	and
6	is	=	use	=	1	than	1	This	answering	was	:-	60	
7	70	.	Sport	=	Sport	=	Sport	There	question	brings	answer	750	?
8	off	<<	14	?	Sports	?	Sports	According	explanation	isn	;	78	.-
9	oga	**	13	<<	first	m	first	It	solution	goes	?	72	-
10	65	---	(	…	Sport	[/	Sport	Although	tap	:	!:	70	,
Input	<|bot|>	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	The	answer	is	:	ans_0

Figure 9: Found gold label computation tree in CODI's vocabulary projections, from sample 31 of GSM8k-Aug's filtered test split. The CODI model does not encode numbers from the question in the latent tokens, at least not in a way that is detectable using vocabulary projection. The model answered this question correctly.

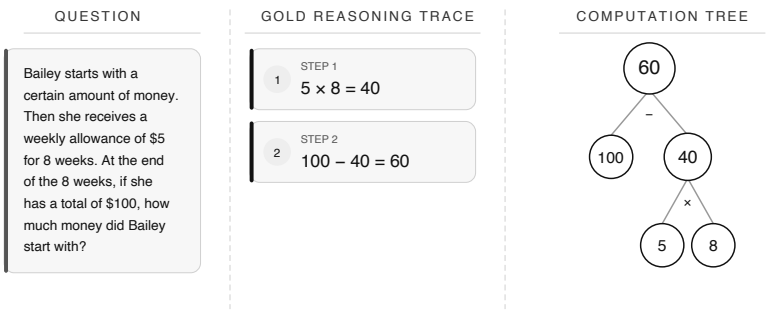


Figure 10: The question, gold reasoning trace, and derived computation tree for sample 63 of GSM8k-Aug's filtered test split.

Output	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0	<|eos|>
1	###	**40**	<|eos|>	###	140	###	###	###	**140**	<|eos|>
2	**5**	**40**	.	**100**	140	<|eos|>	140	140	140	.
3	**8**	50	00	<|eos|>	###	.	140	**100**	540	
4	7	400	<|bot|>		540	What	540	<|eot|>	1440	-
5	10	4	whereas	95	1440	David	1440	60	340	00
6	80	80	50	10	144	washer	340	[illegible]	**60**	$
7	50	44	,	<|bot|>	135	**8**	144	240	120	,
8	**40**	41	80	<|eot|>	60	Initially	139	140	144	?
9	Calcul	400	<|eot|>	10000	160	Neil	160	14	160	What
10	95	20	because	**100**	139		440	64	135	around
Input	<|bot|>	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0

Leaf Operand · Intermediate Result · Final Answer · Step 1 · Incorrect Step 2 · Correct Step 2

Figure 11: Found gold label computation tree in Coconut's vocabulary projections, from sample 63 of GSM8k-Aug's filtered test split. The Coconut model encodes the correct final step, but it encodes an incorrect final step more strongly. The model seems to think that Bailey was losing $5 per week, rather than receiving $5 per week.

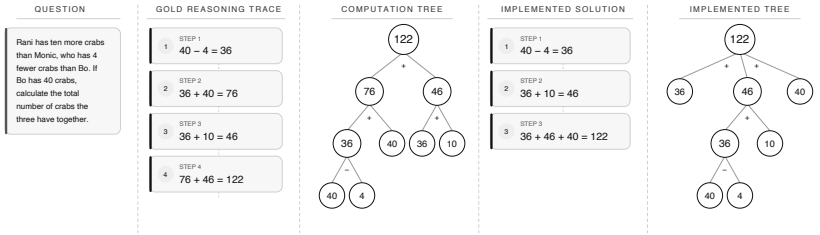


Figure 12: The question, gold reasoning trace, and derived computation tree for sample 160 of GSM8k-Aug's filtered test split.

Output	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0	<|eos|>
1	###	**36**	<|eos|>	###	**46**	###	122	###	**122**	<|eos|>
2	**4**	**36**	.	**10**	**46**	**40**	102	<|eos|>	142	.
3	**40**	**40**	00	<|bot|>	41	<|eot|>	106	122	118	
4	44	35	<|bot|>	5	56	athlon	126	98	126	34
5	<|bot|>	361	2	20	26	Four	136	118	132	18
6	<|eot|>	366	/	<|eot|>	50	aleb	142	Tim	102	-
7	Four	44	###	30	86	Together	142	150	162	14
8	7	156	),	50	71	Tickets	110	Emily	146	14
9	Billy	16	"],	25	26	oby	114	127	136	?
10	0	136	50	15	44	Brian	132	14	130	22
Input	<|bot|>	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0

Leaf Operand Intermediate Result Final Answer Step 1 Step 2 Step 3

Figure 13: Coconut's vocabulary projections, from sample 160 of GSM8k-Aug's filtered test split. The Coconut model encodes a valid reasoning trace not contained in the set of known gold reasoning traces.

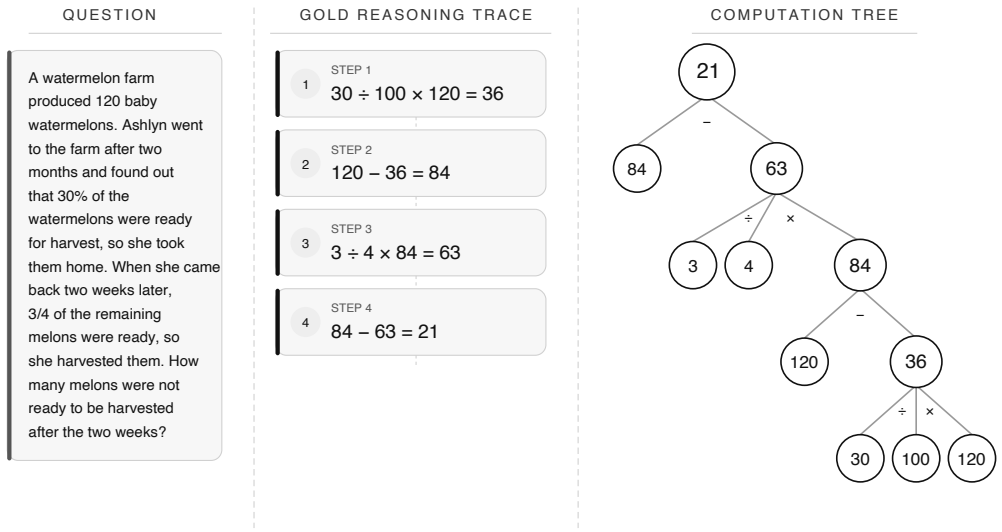


Figure 14: The question, gold reasoning trace, and derived computation tree for sample 207 of GSM8k-Aug's filtered test split.

Output	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0	<|eos|>
1	###	36	<|eos|>	###	84	###	63	###	21	<|eos|>
2	30	36	.	57	84	<|eos|>	63	7	25	.
3	30	42	00	56	840	.	51	21	24	12
4	.	84	###	<|eot|>	204	Tony	61	30	28	6
5	3	18	<|bot|>	36	96	Each	54	31	30	3
6	0	72	whereas	7	168	ml	59	28	26	6
7	90	72	000	.	94	/	66	21	29	84
8	<|bot|>	32	<|eot|>	30	78	Typically	52	25	63	His
9	60	360	2	120	104	3	60	<|eos|>	27	18
10	120	###	6	<|eos|>	88	Three	57	29	42	jeans
Input	<|bot|>	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0

Figure 15: Coconut's vocabulary projections, from sample 207 of GSM8k-Aug's filtered test split. The model seems to encode the percentage 30% as simply 30, instead of 30/100 as in the gold reasoning trace, causing the backtracking search to fail. The arrows indicate what the computation tree looks like when assuming the 30 is representing 30%.

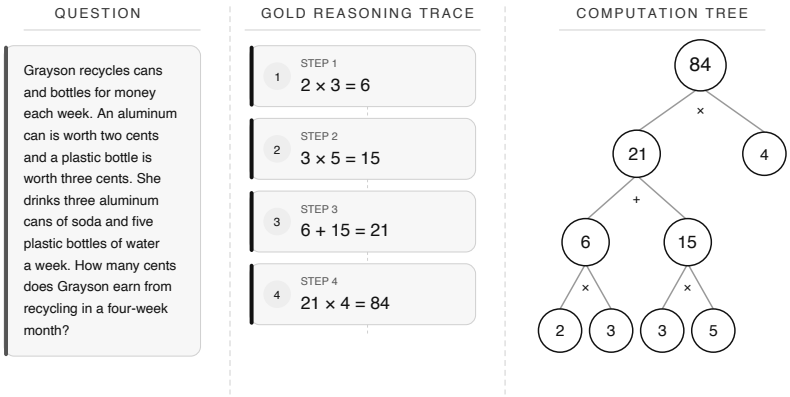
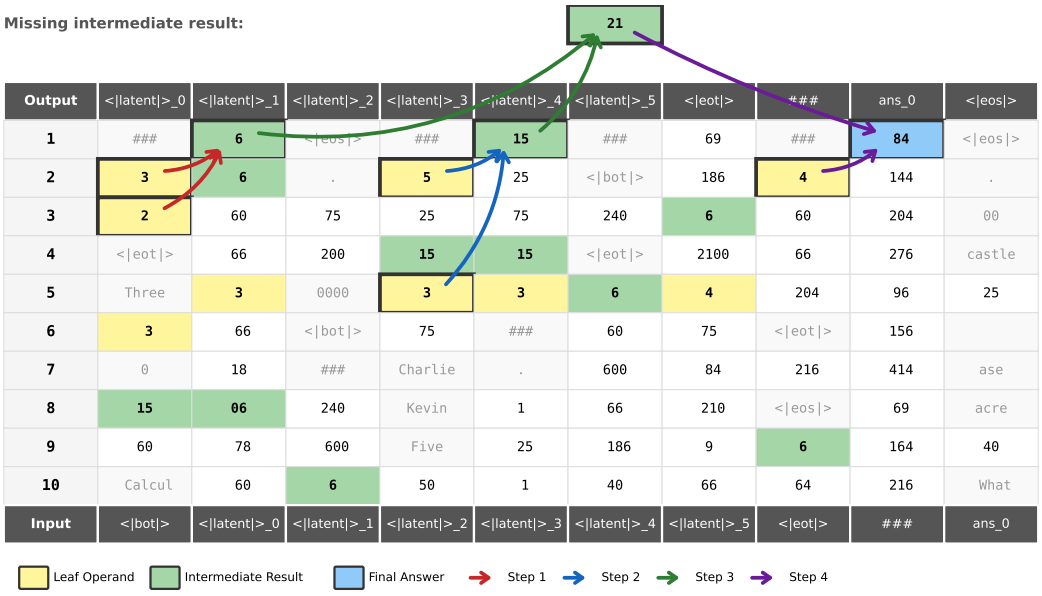


Figure 16: The question, gold reasoning trace, and derived computation tree for sample 415 of GSM8k-Aug's filtered test split.



Output	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0	<|eos|>
1	###	6	<|eos|>	###	15	###	69	###	84	<|eos|>
2	3	6	.	5	25	<|bot|>	186	4	144	.
3	2	60	75	25	75	240	6	60	204	00
4	<|eot|>	66	200	15	15	<|eot|>	2100	66	276	castle
5	Three	3	0000	3	3	6	4	204	96	25
6	3	66	<|bot|>	75	###	60	75	<|eot|>	156	
7	0	18	###	Charlie	.	600	84	216	414	ase
8	15	06	240	Kevin	1	66	210	<|eos|>	69	acre
9	60	78	600	Five	25	186	9	6	164	40
10	Calcul	60	6	50	1	40	66	64	216	What
Input	<|bot|>	<|latent|>_0	<|latent|>_1	<|latent|>_2	<|latent|>_3	<|latent|>_4	<|latent|>_5	<|eot|>	###	ans_0

Figure 17: Coconut's vocabulary projections, from sample 415 of GSM8k-Aug's filtered test split. The gold reasoning trace is encoded, except for the intermediate result 21, which causes the backtracking search to fail.

Table 2: Dataset statistics

Dataset	Training	Validation	Test (original)	Test (filtered)
GSM8k-Aug	385620	500	1319	1194
PrOntoQA	9000	200	800	-
ProsQA	17886	300	500	-

Table 3: Incorrect predictions on GSM8k-Aug where the correct answer appears in the top-10 predicted tokens

Model	Incorrect Samples	Correct Answer in Top-10	Percent
Coconut	793	369	46.5%
CODI	675	337	49.9%