

Knowing Before Saying: LLM Representations Encode Information About Chain-of-Thought Success Before Completion

Anum Afzal

Technical University of Munich
anum.afzal@tum.de

Gal Chechik

Nvidia Research & Bar-Ilan University
gchechik@nvidia.com

Florian Matthes

Technical University of Munich
matthes@tum.de

Yftah Ziser

Nvidia Research
yziser@nvidia.com

Abstract

We investigate whether the success of a zero-shot Chain-of-Thought (CoT) process can be predicted before completion. We discover that a probing classifier, based on LLM representations, performs well *even before a single token is generated*, suggesting that crucial information about the reasoning process is already present in the initial steps representations. In contrast, a strong BERT-based baseline, which relies solely on the generated tokens, performs worse—likely because it depends on shallow linguistic cues rather than deeper reasoning dynamics. Surprisingly, using later reasoning steps does not always improve classification. When additional context is unhelpful, earlier representations resemble later ones more, suggesting LLMs encode key information early. This implies reasoning can often stop early without loss. To test this, we conduct early stopping experiments, showing that truncating CoT reasoning still improves performance over not using CoT at all, though a gap remains compared to full reasoning. However, approaches like supervised learning or reinforcement learning designed to shorten CoT chains could leverage our classifier’s guidance to identify when early stopping is effective. Our findings provide insights that may support such methods, helping to optimize CoT’s efficiency while preserving its benefits.¹

1 Introduction

Chain-of-Thought (CoT) prompting (Wei et al., 2023) enhances the capability of large language models (LLMs) to perform multi-step reasoning. It explicitly guides the LLM in creating intermediate explanations to solve a problem, offering a sequence of reasoning steps while responding to a prompt. Given its effectiveness, CoT has found success in mathematical reasoning (Zheng et al., 2023),

¹Code and data is available at github.com/anum94/CoTpred.

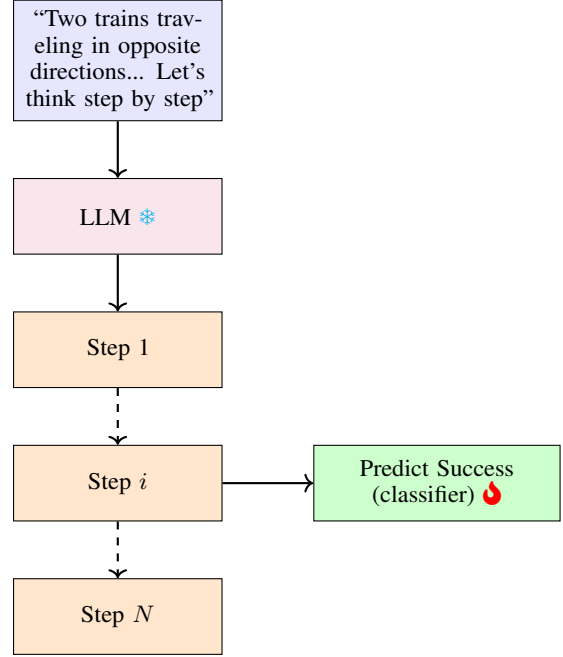


Figure 1: Illustration of our approach. The LLM generates intermediate reasoning steps in a Chain-of-Thought sequence. At step i , we use its internal representations to predict whether the CoT process will succeed. The snowflake (*) indicates frozen parameters, while the flame (🔥) indicates trainable parameters.

medical applications (Liu et al., 2024a), faithfulness evaluation (Xu et al., 2024b), and multimodal models (Wang et al., 2024; Kumari et al., 2024; Byun et al., 2024). While CoT reasoning has been shown to improve performance across many tasks, it is computationally expensive, as it requires decomposing complex problems into a series of intermediate steps, each demanding its own processing. This raises two intriguing questions: a) Do LLMs implicitly "know" whether they will arrive at a correct answer before completing their reasoning? and b) If progressing past the initial steps doesn't improve this knowledge, does this indicate that the LLM has completed its calculation? Given the high computational cost of CoT, understanding

when and how LLMs "know" their answer could enable more efficient and targeted reasoning strategies. Developing a method to assess whether CoT will lead to a correct conclusion could optimize resource allocation—stopping reasoning early when the outcome is clear or dedicating more steps when uncertainty remains. Furthermore, this knowledge could inform annotation efforts to support CoT-specific fine-tuning.

To explore these questions, we create a CoT success prediction dataset derived from popular math-solving datasets, where zero-shot CoT has been shown to significantly outperform vanilla prompting (more details are provided in Section 4.3). By applying a CoT prompt to these questions, we capture the LLM activations across multiple CoT steps and annotate the final answer as either correct or incorrect. We then train a probing classifier (Be-linkov, 2022) on top of the LLM representations (assuming white-box access) to predict if a given prefix of the ongoing CoT sequence would lead to a correct answer (see Figure 1). Our experiments show that by leveraging the LLM’s internal representations, our classifier can effectively predict whether a CoT sequence will be successful—even before generating a single token—achieving 60% to 76.4% accuracy across different datasets and LLMs. Notably, it outperforms BERT (Devlin et al., 2019), a strong text classifier that relies only on the input text, demonstrating that the LLM’s internal representations encode valuable information about intermediate calculations—information that BERT, constrained to shallow linguistic features, cannot capture.

Further experiments with mid-CoT steps reveal an intriguing pattern: in two of six cases, providing the classifier with later reasoning steps does not significantly improve its prediction accuracy. Using SVCCA, a complementary method to probing, we find that in these scenarios, earlier steps are more similar to the final step compared to the dataset where additional CoT context benefits the classifier. We conduct initial zero-shot experiments to investigate whether this similarity allows for early termination of the CoT process without affecting the final answer. While zero-shot alone is insufficient, our results suggest that more targeted approaches, such as supervised learning or reinforcement learning, could effectively shorten the CoT process while maintaining strong performance.

To conclude, our contributions are as follows:

1. We define the task of Chain-of-Thought (CoT) success prediction and investigate whether LLMs inherently estimate the effectiveness of CoT prompting before generating a full answer.
2. We construct datasets for this task and train a lightweight probe that predicts the success of CoT prompting before the LLM completes its generation.
3. Through extensive analysis, we demonstrate that leveraging LLM internal representations significantly improves classification performance compared to relying solely on generated tokens, indicating that these representations capture knowledge about intermediate calculations.
4. We conduct initial experiments on zero-shot early stopping in CoT, showing that while a gap remains between early stopping and full CoT completion, stopping mid-calculation still slightly outperforms not using CoT at all. This suggests that stronger methods could further unlock LLMs’ potential to shorten CoT chains while maintaining high performance.

2 Related Work

We review studies that analyze and customize CoT reasoning and those that use the internal LLM representations to predict aspects of their generation in advance. Given the breadth of research in these areas, we focus on the most relevant works for our setup.

2.1 Analyzing Chain-of-Thought

As CoT reasoning gained popularity, an increasing number of papers sought to analyze its underlying mechanisms, mainly when applied to solving math questions, where it often excels. Xu et al. (2024a) demonstrated how minor changes in numbers or units can drastically affect CoT performance. Several empirical studies have explored key factors in improving CoT performance. For example, Madaan and Yazdanbakhsh (2022) used counterfactual prompts to highlight the importance of symbolic reasoning in the CoT process. Wang et al. (2023a) found that the order of rationales and their relevance to the query are the most crucial aspects of CoT. Rai and Yao (2024) analyzed the neurons in LLM feed-forward layers to determine whether information about the design decisions studied in

empirical research is encoded within them. Alternatively, Liu et al. (2024b) draw a comparison between an LLM and a human’s overthinking nature. They show a series of tasks where similar to humans, LLM’s performance also gets worse with CoT prompting.

Wang et al. (2023a) evaluate which factors play a role in CoT prompting and show that even using incorrect demonstration in CoT prompting leads to the generation of correct answers. In contrast, Cui et al. (2024) show that errors in intermediate reasoning steps tend to affect CoT performance. Lastly, Pfau et al. (2024) used filler words like “...” to show that the CoTs displayed by LLM are just superficial and rather similar internal compute responsible for LLM’s reasoning could be triggered by meaningless filler words. Bao et al. (2025) use structural causal models to analyze how instructions, reasoning steps, and answers interact in CoT prompting. Instead, our work probes internal representations to assess whether models encode information about answer correctness during the reasoning process. While their analysis is causal and text-level, ours focuses on what the model “knows” internally.

2.2 Probing LLMs Internal Representations

The internal representations of LLM have been used to gather insights about tasks; one such task is using representations at a given state t to predict the words at positions beyond $t + 2$ (Goyal et al., 2024). Turpin et al. (2023) try to evaluate if LLM are honest in their explanations and concluded that LLM explanations are heavily biased by simple variations in the prompt, such as reordering of items. Azaria and Mitchell, 2023; Gottesman and Geva, 2024; Seo et al., 2025 investigate the task of estimating LLM’s knowledge on a given subject before it starts generation. They approach this task by using the internal representations of LLM as training features for a probe that can predict if the LLM output would be faithful. The field of probing to comprehend the internal mechanisms of CoT using LLM’s internal representation is still evolving.

2.3 CoT Reasoning for Math Tasks

Recent research has demonstrated the effectiveness of CoT reasoning in logical tasks (Sprague et al., 2024), particularly for solving mathematical datasets. Ahn et al. (2024) examine the latest advancements in large language models (LLMs) for mathematical reasoning and highlight the ef-

fectiveness of CoT in this domain. Furthermore, Ji et al. (2025) introduced a dual CoT approach that incorporates self-reasoning and self-criticism to improve performance on math tasks. Several other studies (Wang et al., 2023b; Li et al., 2023) have focused on improving LLM performance in mathematical tasks by developing custom models optimized for benchmark datasets.

3 Methodology

Constructing the Dataset We first generate deterministic outputs to assess whether an LLM inherently “knows” if it can solve a task using CoT prompting. We run inferences with a temperature of zero, ensuring consistency and eliminating stochastic noise. In addition, low sampling temperatures are recommended for tasks that require precision and factual accuracy, such as technical writing, code generation, or question answering, which is particularly crucial for solving math problems (Renze, 2024). Each generated response is compared to a reference answer, assigning a correctness label that serves as a ground-truth label for our trained classifier. Next, assuming white-box access to the LLM, we extract the LLM’s hidden states from the initial forward pass of the prompt to examine whether its internal representations encode predictive information about CoT success. These hidden states (H) capture the LLM’s pre-generation reasoning and are used as training features. For each sample, we obtain a 3D tensor (H, N, k), where $H = h_1, h_2, \dots, h_L$ represents the hidden layers, N is the number of samples, and k is the hidden dimension size. Since prompt lengths vary, we use the last token’s representation for consistency. To study the contribution of different layers, we train a separate probe for each hidden layer, evaluating its accuracy as a measure of the layer’s role in CoT success prediction. A higher classification accuracy (C_L) indicates that layer L contains more information about the likelihood of success. We conduct our experiments using Llama-3.1-8B and Mistral-7B, utilizing all hidden layers, each with 4096 dimensions.

Classification Model Following the precedent set by Azaria and Mitchell (2023), we employ a compact feedforward neural network with three hidden layers of 256, 128, and 64 units, each using ReLU activation. The output layer applies a sigmoid activation function. Training is optimized using either the Adam or SGD optimizer, which is

selected based on empirical performance for each dataset. We perform a hyperparameter tuning specific to the task and detail the selected configurations in [Appendix A](#). The classifier is trained for five epochs across all datasets and LLMs.

Success Prediction over Time While our primary methodology focuses on predicting CoT performance before LLM generation begins, we also explore how this prediction may change during the generation process. Identifying atomic steps in the generation can be challenging due to their varying lengths and styles, so for simplicity, we instead use percentages of the total number of tokens generated (10%, 20%, up to 90%) and concatenate them with the initial prompt (see [Appendix C](#) for illustration). This approach allows for a consistent evaluation across different generations. As in the previous approach, we extract hidden layers for partial generations at 10% intervals, enabling us to observe the evolution of internal representations. We apply a similar strategy to expand our test sets.

CoT	wo CoT	success rate		
		Cn-k12	AQuA	Olympiad
Llama-3.1-8B				
0	0	41.83%	47.46%	44.76%
0	1	8.11%	2.53%	4.77%
1	0	32.52%	43.32%	36.97%
1	1	17.44%	6.67%	12.92%
Mistral-7B				
0	0	44.07%	39.43%	41.43%
0	1	5.89%	10.54%	8.46%
1	0	32.33%	33.83%	28.08%
1	1	17.69%	16.2%	22.03%

Table 1: We depict the percentage of Problems the LLM was able to solve with and without any Chain-of-Thought prompting on a balanced dataset where CoT prompting helped solve 50% of the problems. We show a confusion matrix such that 0 means LLM was not able to solve the problem, and 1 means that it was able to solve the task.

4 Experimental Setup

4.1 Baseline

The prediction of CoT success may depend more on linguistic cues in the text than on internal LLM representations encoding intermediate computations. To explore this, we build on ([Azaria and Mitchell, 2023](#)), which examined whether LLMs

store information in their internal states, albeit for different purposes. Our goal is to demonstrate that such information is indeed retained within the LLM’s representations. To test this, we use BERT as a baseline, as it relies solely on textual tokens without access to internal LLM states, effectively functioning as a black-box access approach. Given BERT’s strength as a text classifier, its ability to predict CoT success would indicate that textual cues alone suffice. We use the google-bert/bert-base-uncased variant with default settings, maintaining a consistent neural network-based classification setup while varying the input features.²

4.2 Large Language Models

For the hidden representations of Llama-3.1-8B and Mistral-7B, we use the meta-llama/Llama-3.1-8B-Instruct and mistralai/Mistral-7B-Instruct-v0.3 checkpoints on huggingface respectively.

4.3 Datasets

We used three math problem datasets of varying difficulty in our experiments: World Olympiads Data (Olympiad) ([LI et al., 2024](#)), Chinese K-12 Exam (cn-k12) ([LI et al., 2024](#)), and AQuA (aqua) ([Ling et al., 2017](#)). To assess model behavior across different reasoning patterns, we ran each dataset with two different LLMs—Llama-3.1-8B ([Grattafiori et al., 2024](#)) and Mistral-7B—resulting in six distinct dataset variants. Since we enforce class balance (i.e., an equal number of correct and incorrect generations), the two versions of the same original dataset may contain different sets of questions, depending on the LLM’s outputs. Given the difficulty level of questions within each dataset, the success rate using Llama-3.1-8B, for example, varies considerably, reaching 22%, 28%, and 62.3% on Olympiad, cn-k12, and AQuA, respectively. For all datasets and both LLMs, we ran inference to obtain balanced train (10k) and test (1k) sets with an equal distribution of positive and negative examples. Consequently, our classification model is trained on a 10000×4096 feature space for each dataset and LLM layer combination. We reserve 10% of the training set for validation to tune hyperparameters. The distribution of question and generation lengths is summarized in [Table 2](#). Lastly, it is also important to evaluate how well the same

²Training features vary when using BERT embeddings (dimension 768) or LLM layers (dimension 4096).

Dataset	# Question Tokens			# Generation Tokens		
	avg	min	max	avg	min	max
Llama-3.1-8B						
AQuA	42.96	5	308	287.24	17	512
Olympiad	67.76	3	1109	401.56	2	512
Cn-k12	76.38	7	520	327.46	4	512
Mistral-7B						
AQuA	84.23	38	344	255.07	35	509
Olympiad	86.10	9	1108	368.47	40	511
Cn-k12	88.94	5	679	304.96	29	511

Table 2: Average, Minimum and Maximum token count of Questions and LLM generation using Llama-3.1-8B and Mistral-7B tokenizers on all three datasets. Generation Tokens are capped at 512 using max_new_tokens = 512.

LLM performs when solving the same problems without chain-of-thought (CoT) prompting. We present this comparison in a confusion matrix in Table 1, which shows that, for instance, without CoT, Llama-3.1-8B achieved only 9.2%, 17.69%, and 25.55% success rates on AQuA, Olympiad, and cn-k12, respectively, substantially lower than the 50% CoT accuracy we enforce when collecting a balanced dataset.

Olympiad: The Math Olympiad is a competitive examination designed to evaluate students’ mathematical skills and competencies. We use the olympiad dataset from the NuminaMath-CoT (LI et al., 2024) collection of the dataset. This dataset is a collection of problems and respective answers following a CoT format collected from international/national contests as well as forums, books, and summer school materials.

Cn-k12: This large-scale Chinese K-12 education math exercise dataset was translated and re-aligned to English using GPT-4. This dataset is also part of the NuminaMath-CoT (LI et al., 2024) collection of datasets, and reference answers follow a CoT format.

AQuA: This dataset (Ling et al., 2017) consists of algebraic math problems, each presented with a step-by-step reference solution. Unlike the other two datasets, it includes multiple-choice options. We use the original dataset released by the authors.

4.4 Manual Annotation

Given the large dataset size, we used GPT-4o mini³ to label the training and validation sets for

³<https://platform.openai.com/docs/models#gpt-4o-mini>

the first language model (Llama-3.1-8B). To ensure the reliability of our evaluation, we manually annotated the test set using annotators selected from a local university’s STEM Master’s program, compensated at 16 euros per hour. Annotators evaluated the correctness of each generation based on the question and a reference answer. On a test set of 1,000 samples, human annotations agreed with LLM-generated ones on 90.9%, 94.8%, and 93.4% for AQuA, Cn-K12, and Olympiad, respectively. These numbers indicate that although LLM-generated labels are generally reliable, the disagreement with human judgment suggests caution when using them for evaluation. For the second language model (Mistral-7B), all datasets (training, validation, and test) were annotated using GPT-4o mini. Given the imperfect agreement observed between human and GPT-4o mini annotations for the first model, these results should be interpreted with appropriate caution.

5 Results and Discussion

Since the test set for Llama-3.1-8B was manually annotated by human experts, it is considered more reliable than the GPT-4o mini-annotated test set used for Mistral-7B. As a result, in certain analyses where only one LLM is presented, we focus solely on Llama-3.1-8B to ensure more reliable evaluation.

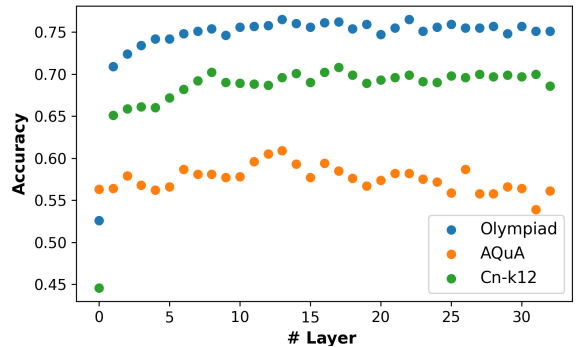


Figure 2: Accuracy on the test set when the probe is trained and tested on hidden representations from each layer. Results are shown for all 33 layers of Llama 3.1 8B Instruct across all three datasets.

5.1 Prediction before Generation

Main Results We evaluate the information contained in the internal representation of the LLM before it begins generating and present the results in Table 3, specifically under the %0 column. Given that 50% of answers in the dataset are correct (bal-

Dataset	Before Gen		Over Time									
	top-5 layers	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Llama-3.1-8B (human-annotated)												
<i>BERT (baseline)</i>												
AQuA	-	53.50	54.2	55.8	54.1	54.1	53.5	51.7	51.6	53.7	51.7	50.6
Olympiad	-	69.10	67.9	68.7	66.2	66.4	67.4	65.1	67.9	66.8	67.1	66.9
Cn-k12	-	66.20	57.8	64.4	64.1	63.8	64.3	63.8	62.7	61.7	61.1	62.4
<i>Our Model</i>												
AQuA	11, 12, 13, 14, 16	60	51.5	60	60.5	63.2	60.11	60.8	63.7	62.9	65.7	69.4
Olympiad	8, 14, 16 , 17, 31	76.4	75.1	75.8	73.8	74.6	76.5	73.8	75.3	75.3	73.8	75.9
Cn-k12	13, 14, 16 , 17, 22	69.10	67.7	69.9	69.2	67.7	67.2	70	71.6	70.7	68.9	70.9
Mistral-7B (GPT-4o mini annotated)												
<i>BERT (baseline)</i>												
AQuA	-	60.1	57.8	59.5	63.4	65.2	63.7	67.8	66.7	66.8	68.1	71.8
Olympiad	-	68.8	68.1	69.6	68.8	68.3	68.6	66.4	67.1	68.3	68.6	69.6
Cn-k12	-	65.5	63.5	65.8	65.1	64.4	64.8	66.5	66.7	67.3	67.8	68.5
<i>Our Model</i>												
AQuA	15, 16, 18 , 23, 28	64.7	54.4	66.1	66.9	66.4	65.1	65.8	64.1	66.6	67.4	80.2
Olympiad	7, 9, 18 , 26, 28	71.8	71.7	72.0	72.3	74.1	74.2	75.6	74.5	75.3	75.5	75.9
Cn-k12	12, 14, 18 , 21, 24	67.1	68.0	68.4	66.7	67.7	67.8	67.1	68.1	68.6	67.6	71.4

Table 3: Classification model accuracy before generation begins and as it progresses, measured at different completion percentages. The best-performing layers and time steps are highlighted in bold.

anced data), a random classifier would achieve an accuracy of 50%, which serves as the baseline for interpreting the results. The BERT baseline, which relies exclusively on token representations on CoT prompt with question, consistently outperformed random chance across all six datasets. On generations from Llama-3.1-8B, BERT achieved accuracy scores ranging from 53.5% to 69.1%. For Mistral-7B, BERT’s accuracy ranged from 60.1% to 68.8%. However, the predictive power appears relatively weaker for the AQuA dataset⁴. Our suggested method, which uses the LLM’s internal representations, outperformed the BERT baseline across all datasets, demonstrating their importance for predicting CoT success before generation begins. For instance, on Llama-3.1-8B datasets, our model achieved 60.0% accuracy on AQuA, significantly higher than BERT, and 76.4% on Olympiad, compared to BERT’s 69.1%. We observe similar patterns for the Mistral-7B datasets. The large variance in results (ranging from 60.0% to 76.4%) suggests that the effectiveness of the internal rep-

resentations may be influenced by factors such as task complexity or dataset characteristics, which warrant further investigation. See Appendix B for details on Classification Performance Breakdown.

LLM Layers Analysis In our experiments, we evaluated the accuracy of the classification model when trained on the hidden representations of each layer. We show the accuracy per layer of Llama-3.1-8B in Figure 2. It can be seen that the middle such as layers 11 - 14 and layers 16 - 17, and in some cases the last layers of the LLM seem to play a role in it’s internal notion of Success or Failure. As the in shown Table 3, we find layer 14 and layer 16 to be consistent among all three datasets suggesting these layers to be more involved in the notion of CoT prediction. Our findings are inline with those of Azaria and Mitchell 2023, who show similar results regarding LLM’s notion of truthfulness and show layer 16 and in some cases the last layers of LLM to be most knowledgeable for their task.

5.2 Prediction over Time

Main Results We evaluate the information contained in the internal representation of the LLM

⁴We hypothesize that accuracy on Llama-3.1-8B-generated AQuA dataset might be lower as the sampled questions might contain less linguistic patterns than the other datasets.

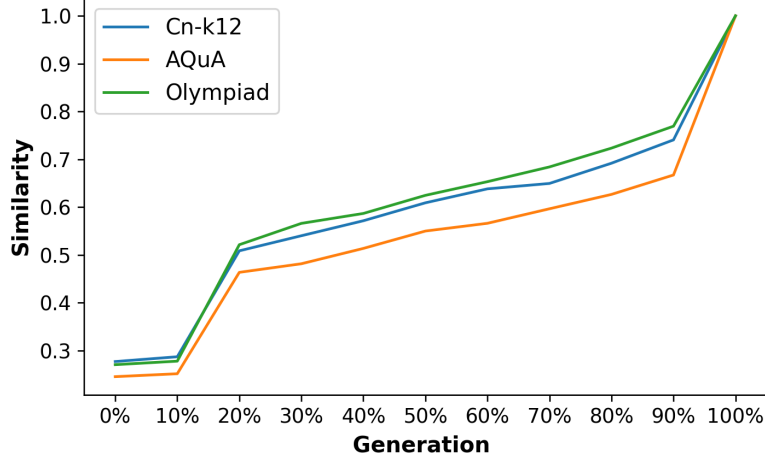


Figure 3: Similarity scores using SVCCA between hidden representations of each time step and the full generation, using Layer 14 for all three Llama-3.1-8B datasets.

during generation, as shown in Table 3. To do this, we capture the hidden states at various time intervals—after the LLM has generated 10%, 20%, and so on of the total content—and use these states to train the probe. The goal is to assess whether the LLM’s understanding of success or failure becomes more apparent as more content is generated. This aligns with the idea that as more information is revealed, the model’s understanding of success or failure becomes clearer, similar to how humans typically gain better insight into a task as they progress further. We selected the layer that performed best on the given dataset in prior evaluations for the prediction-over-time experiments. Interestingly, in the Llama-3.1-8B-generated dataset, BERT’s performance tends to decline as more context is revealed, whereas for Mistral-7B, it remains relatively stable. While BERT can effectively process surface-level cues such as question structure, it appears less capable of tracking the evolving reasoning embedded in the chain-of-thought (CoT) generation. As the CoT unfolds and more tokens are added, the increasing complexity may exceed BERT’s shallow interpretive capacity, limiting its ability to follow deeper logical developments. In contrast, using internal LLM representations as input significantly improves performance—e.g., from 60% to 69.4% on AQuA in the Llama-3.1-8B-generated dataset—and yields consistent gains across all Mistral-7B datasets. For Olympiad and cn-k12 with Llama-3.1-8B, results remain comparable. This demonstrates the representations’ ability to capture more nuanced, cumulative reasoning and better reflect the underlying complexity

of the CoT process, ultimately enabling more accurate success prediction. Surprisingly, Mistral-7B-generated AQuA dataset performs better as compared to the Llama-3.1-8B-generated one. Further inspection of the generations revealed that for the negative examples, Mistral-7B generations often assume that the provided question might be incorrect. These linguistic queues as shown in Table 4, might be used by BERT baseline as a feature to classify negative examples.

Complimentary Analysis for Probing Our probing results suggest that the initial CoT steps in the Olympiad and Cn-K12 Llama-3.1-8B generated datasets contain similar knowledge about CoT success as the later stages, which may indicate that they share an understanding of the calculations performed up to that point. However, prior research has highlighted several limitations of probing. For instance, when trained on a large number of examples, models using randomly initialized embeddings as input representations can outperform those using pretrained intermediate representations (Zhang and Bowman, 2018; Wieting and Kiela, 2019). To complement probing-based analyses, several alternative methods assess knowledge in model representations by measuring their correlations without relying on a classifier. These include SVCCA (Singular Vector Canonical Correlation Analysis) (Raghu et al., 2017), PWCCA (Projection Weighted Canonical Correlation Analysis) (Morcos et al., 2018), and CKA (Centered Kernel Alignment) (Kornblith et al., 2019), among others. Such methods are frequently used as alternatives for probing in NLP research (Saphra and

Example 1

1. First, we need to find the probability that Kyle and Catherine solve the problem but David does not.
2. The probability that Kyle solves the problem is $1/3$, and the probability that Catherine solves the problem is $5/9$.
3. The probability that David does not solve the problem is $1 - 2/7 = 5/7$.
4. To find the probability that all three events happen, we multiply their individual probabilities: $(1/3) * (5/9) * (5/7)$.
5. Simplify the expression: $(5/27) * (5/7) = 25/189$.
6. **The answer is E) 20 / 189. However, there seems to be a mistake in the options provided. The correct answer should be 25/189, not 20/189. So, the correct answer is E) 25 / 189.**

Example 2

1. The total investment is Rs. 70000 (A) + Rs. 120000 (B) = Rs. 190000.
2. The total profit is Rs. 26000.
3. To find the share of B, we need to divide the total profit by the total investment and then multiply it by B's investment.
Share of B = (Total profit / Total investment) * B's investment
= $(26000 / 190000) * 120000$
= $(26/19) * 12000$
= $24000 * (26/19)$
= $24000 * (1.3158)$
= 24009
So, the share of B is approximately Rs. 24009.
However, since the options provided are not exact, we can round the answer to the nearest option. The closest option is D) 24009. But since the question asks for one of the options, we cannot be certain that D) 24009 is the correct answer. It's a tricky question due to the rounding of the options.
Final answer: The share of B is approximately 24009, but the question asks for one of the options, so the exact answer cannot be determined from the given options.

Table 4: Linguistic Patterns in the negative examples of Mistral-7B-generated AQUA dataset, which might be used by BERT classifier as a feature in our Prediction over Time experiments.

Lopez, 2018; Zhao et al., 2023, 2024). In our study, we employ SVCCA following Zhao et al. (2022), which investigated how domain-specific information is encoded in LLMs—a research focus related to ours.

We compute SVCCA scores between representations at each reasoning step⁵ (10%, 20%, ..., 90%) with 100% (CoT completion), where higher scores indicate greater similarity in encoded information (Figure 3). Since the AQUA dataset consists of shorter questions and CoT sequences (see Table 2), one might expect its intermediate representations to be more similar across steps, as the reasoning process is more compact. However, we observe that AQUA exhibits lower SVCCA scores between earlier and later steps compared to Olympiad and Cn-K12. This strengthens our probing findings: when earlier steps contain information predictive of CoT success, representations remain more stable throughout the reasoning process. This suggests that intermediate representations may not only encode predictive information about the final answer correctness but also the final answer itself. If the model is implicitly performing parts of the final computation at earlier steps, we may be able to leverage this by directly prompting it to provide an answer before completing the full chain of thought.

⁵See Appendix B for scores between all possible combinations of representations reasoning step

Early Stopping in CoT Reasoning To examine whether the model’s intermediate representations encode sufficient information for correct answers, we prompt it to generate an answer at various reasoning steps and evaluate its accuracy. As in our previous experiments, this intervention is performed in a zero-shot manner without explicitly training the model to follow such instructions. Specifically, we halt the Llama-3.1-8B reasoning process by providing the generated sequence up to a certain point, followed by the instruction: *"Stop all computation and give me the correct answer in 2–3 words, if you already know it"*. This allows us to assess whether the model can extract a final answer without completing the full chain of thought (See Appendix B for an illustration). We conduct human annotation on 100 samples per dataset at three timesteps⁶, summarizing the results in Table 5. Our analysis examines how often the halted response remains consistent with the final, uninterrupted answer, how often it changes (inconsistent), and in what fraction of those cases the change leads to a corrected answer—where stopping CoT reasoning actually improves performance. Finally, we compare overall correctness rates to both the full CoT process and a setting without CoT. The relatively low consistency rate, even at 99% comple-

⁶We select two intervals where the classification model achieves its highest accuracy and one at the final step.

Dataset	Gen %	Consistent	Inconsistent	Corrected	Correct	W_CoT_Correct
AQuA	50	40	45	15	37	38
	70	48	41	11	37	
	99	81	14	5	59	
Olympiad	30	21	76	3	18	19
	50	28	70	2	22	
	99	57	41	2	33	
Cn-k12	30	37	55	7	32	24
	50	42	49	7	33	
	99	88	9	3	47	

Table 5: LLM’s generation was artificially interrupted to halt computation and were asked to just provide their best guess on 100 samples at 3 different time steps defined Gen. Annotators marked an answer Consistent if it is same the answer it provides when it is allowed to continue generation and Inconsistent if the provided answer differ. There were cases where full generation led to incorrect solutions by LLM and these interruptions allowed LLM to generate the correct answer, which was given the label Correct by annotators.

tion—particularly in Olympiad (57%)—suggests that zero-shot early stopping is a suboptimal intervention. The model does not always converge to a stable answer, even when nearly the entire reasoning sequence is generated, highlighting the limitations of simply prompting it to stop early. However, despite this brittleness, halting CoT midway in AQuA and Cn-K12—the two datasets where later reasoning steps did not enhance CoT success predictability—still slightly outperforms the setting without CoT. This indicates that even incomplete CoT sequences can carry enough information to improve accuracy, revealing untapped potential in intermediate reasoning states. These findings suggest that while zero-shot interventions have limitations, more sophisticated approaches—such as training the model to generate concise reasoning chains through supervised learning or reinforcing brevity via RL-based rewards—could more effectively unlock this potential. By optimizing the model’s ability to extract key reasoning steps without unnecessary verbosity, future methods could further bridge the gap between full CoT and early stopping while maintaining or even improving accuracy.⁷

6 Conclusion

We demonstrate that the success of the CoT reasoning process can be predicted from the internal representations of the LLM even before the generation of a single token. However, we also observe

that, in some cases, the accuracy of this prediction does not improve when the classifier is exposed to intermediate reasoning steps. Using SVCCA, we show that early steps encode information that is more similar to the final steps in these cases. This raises the question of whether these early representations also contain valuable information about the final answer itself. Our initial experiments suggest that while this potential exists, zero-shot prompting may not fully unlock it. We hope our findings will inform future research aimed at making CoT more efficient without sacrificing accuracy, as the computational cost of CoT is significant.

Limitation

Manually annotating test examples for each dataset limits the generalizability of our study, as we use human evaluation for only a single LLM and focus on three math datasets in a zero-shot setting. Additionally, we use a temperature of zero to minimize stochastic noise in the generation process, which could accumulate over multiple reasoning steps. However, this setting may not fully capture the variability present in real-world, stochastic LLM usage. Furthermore, our method assumes white-box access to the model, which is not typically available for proprietary models.

Ethics Statement

Our research focused on evaluating the internal representation of a Large Language Model to better the notion of prediction for CoT Reasoning. During out research, we didn’t perform any fine-tuning

⁷In some cases, halting CoT before completion improved answer correctness, as reflected in the *corrected* column in the table.

that could introduce any bias in the LLM. We work with open-source datasets and hence produce no additional bias than what might already be part of it. During our manual evaluations, annotators were asked to correct math problems, where there is only one logically correct answer, reducing the suspect of biased annotations. We mostly used existing algorithms for supporting our analysis, hence making our findings more reliable. For reproducibility, we release the hyper-parameters used in our experiments.

References

- Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. [Large language models for mathematical reasoning: Progresses and challenges](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, St. Julian’s, Malta. Association for Computational Linguistics.
- Amos Azaria and Tom Mitchell. 2023. [The internal state of an LLM knows when it’s lying](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976, Singapore. Association for Computational Linguistics.
- Guangsheng Bao, Hongbo Zhang, Cunxiang Wang, Linyi Yang, and Yue Zhang. 2025. [How likely do LLMs with CoT mimic human reasoning?](#) In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7831–7850, Abu Dhabi, UAE. Association for Computational Linguistics.
- Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.
- Ju-Seung Byun, Jiyun Chun, Jihyung Kil, and Andrew Perrault. 2024. [ARES: Alternating reinforcement learning and supervised fine-tuning for enhanced multi-modal chain-of-thought reasoning through diverse AI feedback](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4410–4430, Miami, Florida, USA. Association for Computational Linguistics.
- Yingqian Cui, Pengfei He, Xianfeng Tang, Qi He, Chen Luo, Jiliang Tang, and Yue Xing. 2024. [A theoretical understanding of chain-of-thought: Coherent reasoning and error-aware demonstration](#). *Preprint*, arXiv:2410.16540.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Daniela Gottesman and Mor Geva. 2024. [Estimating knowledge in large language models without generating a single token](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3994–4019, Miami, Florida, USA. Association for Computational Linguistics.
- Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. [Think before you speak: Training language models with pause tokens](#). *Preprint*, arXiv:2310.02226.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Alonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Is-han Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelier van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yearly, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kam-badur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick

Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Rapparthi, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collet, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Conguet, Virginie Do, Vish Vogeti, Vitor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyan Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkan Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippas Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim

Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabisa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanachandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuang Zhang, Shuang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.

- Shihao Ji, Zihui Song, Fucheng Zhong, Jisen Jia, Zhaobo Wu, Zheyi Cao, and Tianhao Xu. 2025. [Mygo multiplex cot: A method for self-reflection in large language models via double chain of thought thinking](#). *Preprint*, arXiv:2501.13117.
- Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR.
- Gitanjali Kumari, Kirtan Jain, and Asif Ekbal. 2024. [M3Hop-CoT: Misogynous meme identification with multimodal multi-hop chain-of-thought](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22105–22138, Miami, Florida, USA. Association for Computational Linguistics.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. [Camel: Communicative agents for "mind" exploration of large scale language model society](#). *Preprint*, arXiv:2303.17760.
- Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [<https://huggingface.co/AI-MO/NuminaMath-CoT>](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *ACL*.
- Jiaxiang Liu, Yuan Wang, Jiawei Du, Joey Tianyi Zhou, and Zuozhu Liu. 2024a. [MedCoT: Medical chain of thought via hierarchical expert](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17371–17389, Miami, Florida, USA. Association for Computational Linguistics.
- Ryan Liu, Jiayi Geng, Addison J. Wu, Ilia Sucholutsky, Tania Lombrozo, and Thomas L. Griffiths. 2024b. [Mind your step \(by step\): Chain-of-thought can reduce performance on tasks where thinking makes humans worse](#). *Preprint*, arXiv:2410.21333.
- Aman Madaan and Amir Yazdanbakhsh. 2022. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*.
- Ari Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. *Advances in neural information processing systems*, 31.
- Jacob Pfau, William Merrill, and Samuel R Bowman. 2024. Let’s think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*.
- Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. [Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability](#). *Preprint*, arXiv:1706.05806.
- Daking Rai and Ziyu Yao. 2024. [An investigation of neuron activation as a unified lens to explain chain-of-thought eliciting arithmetic reasoning of LLMs](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7174–7193, Bangkok, Thailand. Association for Computational Linguistics.
- Matthew Renze. 2024. [The effect of sampling temperature on problem solving in large language models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7346–7356, Miami, Florida, USA. Association for Computational Linguistics.
- Naomi Saphra and Adam Lopez. 2018. Understanding learning dynamics of language models with svcca. *arXiv preprint arXiv:1811.00225*.
- Yeongbin Seo, Dongha Lee, and Jinyoung Yeo. 2025. Detecting hallucination before answering: Semantic compression through instruction.
- Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. [To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning](#). *Preprint*, arXiv:2409.12183.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. [Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 74952–74965. Curran Associates, Inc.
- Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023a. [Towards understanding chain-of-thought prompting: An empirical study of what matters](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2717–2739, Toronto, Canada. Association for Computational Linguistics.
- Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2023b. [Mathcoder: Seamless code integration in llms for enhanced mathematical reasoning](#). *Preprint*, arXiv:2310.03731.
- Yan Wang, Yawen Zeng, Jingsheng Zheng, Xiaofen Xing, Jin Xu, and Xiangmin Xu. 2024. [VideoCoT:](#)

A video chain-of-thought dataset with active annotation tool. In *Proceedings of the 3rd Workshop on Advances in Language and Vision Research (ALVR)*, pages 92–101, Bangkok, Thailand. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. [Chain-of-thought prompting elicits reasoning in large language models](#). *Preprint*, arXiv:2201.11903.

John Wieting and Douwe Kiela. 2019. No training required: Exploring random encoders for sentence classification. *arXiv preprint arXiv:1901.10444*.

Ancheng Xu, Minghuan Tan, Lei Wang, Min Yang, and Ruifeng Xu. 2024a. [NUMCoT: Numerals and units of measurement in chain-of-thought reasoning using large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14268–14290, Bangkok, Thailand. Association for Computational Linguistics.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024b. [Faithful logical reasoning via symbolic chain-of-thought](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13326–13365, Bangkok, Thailand. Association for Computational Linguistics.

Kelly W Zhang and Samuel R Bowman. 2018. Language modeling teaches you more syntax than translation does: Lessons learned through auxiliary task analysis. *arXiv preprint arXiv:1809.10040*.

Zheng Zhao, Yftah Ziser, and Shay Cohen. 2022. [Understanding domain learning in language models through subpopulation analysis](#). In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 192–209, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Zheng Zhao, Yftah Ziser, and Shay B Cohen. 2024. [Layer by layer: Uncovering where multi-task learning happens in instruction-tuned large language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15195–15214, Miami, Florida, USA. Association for Computational Linguistics.

Zheng Zhao, Yftah Ziser, Bonnie Webber, and Shay Cohen. 2023. [A joint matrix factorization analysis of multilingual representations](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12764–12783, Singapore. Association for Computational Linguistics.

Mingyu Zheng, Hao Yang, Wenbin Jiang, Zheng Lin, Yajuan Lyu, Qiaoqiao She, and Weiping Wang. 2023. [Chain-of-thought reasoning in tabular language models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11006–11019, Singapore. Association for Computational Linguistics.

A Technical Details

Classification Models We performed hyperparameter-tuning for each dataset (and each layer) when training the classification models. We illustrate the best hyperparameters in [Table 7](#) and used them in our Prediction over time experiments.

B Additional Results

Complimentary Analysis for Probing We compute SVCCA scores between all possible combinations of representations reasoning step (10%, 20%, ..., 90%) and with 100% (CoT completion) and show them in [Figure 4](#).

Early Stopping in CoT Reasoning We show an illustration of Early Stopping in CoT Reasoning and the LLM response in [Table 8](#). It can be seen that despite being prompted to stop calculations to 30% of it’s original generation, it still generates the correct answer.

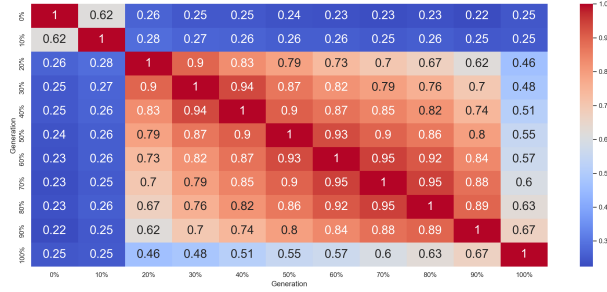
Classification Performance Breakdown Following our earlier evaluations, we perform an in-depth analysis of the classification model’s performance on a test set of 1,000 examples, summarized in [Table 9](#). Unlike the BERT baseline, which performs well on true positives (TP) only, our model excels at identifying both true negatives (TN) and true positives (TP). For example, on the Llama-3.1-8b generated Cn-k12 dataset, BERT identified 250 true negatives, while our model identified 317. Similarly, on AQuA, BERT’s true negatives were 131, while our model achieved 177, and on Olympiad, BERT had 315 true negatives, compared to 388 for our method. These results show that our model not only performs well on true positives but also significantly outperforms BERT in detecting true negatives. This suggests that our method captures

Cn-k12	AQuA	Olympiad
layer 31	layer 32	layer 13
layer 10	layer 10	layer 28
layer 11	layer 13	layer 32
layer 13	layer 12	layer 10
layer 17	layer 5	layer 3

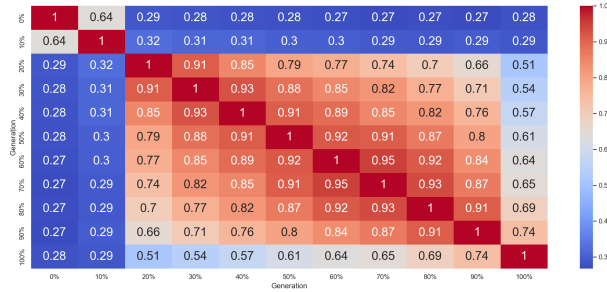
Table 6: Five most dissimilar layers using SVCCA (Singular Vector Canonical Correlation Analysis) between hidden representation with and with CoT prompt per layer, averaged over 1000 samples.

Dataset	batch size	weight init	learning rate	optimizer	threshold
<i>BERT (baseline)</i>					
AQuA	128	HE_uniform	0.001	sgd	0.6
Olympiad	128	HE_uniform	0.001	adam	0.6
Cn-k12	128	-	0.001	adam	0.6
<i>Our Model</i>					
AQuA	32	HE_uniform	0.001	adam	0.5
Olympiad	128	HE_normal	0.001	adam	0.5
Cn-k12	16	HE_uniform	0.001	sgd	0.5

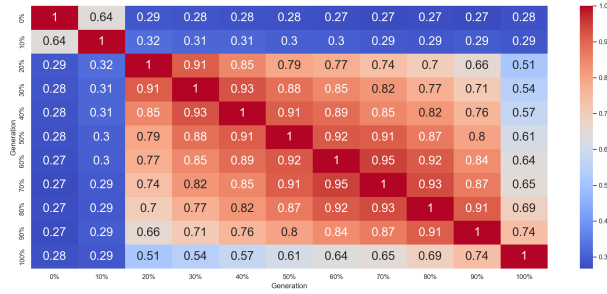
Table 7: Accuracy of the classification model before it starts generating and over time when model has generated x% of the answer.



(a) AQuA



(b) Cn-k12



(c) Olympiad

Figure 4: Similarity Scores using Singular Vector Canonical Correlation Analysis (SVCCA) of the internal representation of LLM through Prediction over Time using Layer 14.

A high school has a total of 900 students, among which there are 300 freshmen, 200 sophomores, and 400 juniors. Now, a stratified sampling method is used to select a sample of 45 students. How many students should be selected from each grade, respectively?

A: 15, 5, 25

B: 15, 15, 15

C: 10, 5, 30

D: 15, 10, 20

Let's think step by step:

To determine the number of students to be selected from each grade, we need to calculate the proportion of students in each grade and then apply this proportion to the total sample size of 45 students.

Step 1: Calculate the proportion of students in each grade

Stop all computation and give me the correct answer in 2- 3 words, if you know it already.

Answer **15, 10, 20**

Reference Answer The correct answer is D.

Table 8: An Illustration of a sample showing how we artificially halt LLM generation when it has generated 30% of the answer.

more relevant predictive signals, enabling a more nuanced understanding of the factors that drive Chain-of-Thought success.

LLM Layer Analysis We also include the representation of LLM without any CoT reasoning in our experiments. In Table 6, we illustrate the 5 most dissimilar layers. Since they were both encoded using the same problems, the only difference is the inclusion of CoT Reasoning, the dissimilarity among the layers may just play a more notable role in LLM's internal notion of CoT Reasoning.

C Dataset Examples

Success Prediction over Time An illustration of samples from the Success Prediction over Time method is shown in Table 10 for 2 different generation steps.

Cn-k12	AQuA	Olympiad
Llama-3.1-8B		
<i>BERT (baseline)</i>		
250 (TN), 196 (FP) 142 (FN), 412 (TP)	131 (TN), 306 (FP) 159 (FN), 404 (TP)	315 (TN), 159 (FP) 135 (FN), 391 (TP)
<i>Our Model</i>		
317 (TN), 129 (FP) 180 (FN), 374 (TP)	177 (TN), 269 (FP) 147 (FN), 416 (TP)	388 (TN), 86 (FP) 153 (FN), 373 (TP)
Mistral-7B		
<i>BERT (baseline)</i>		
282 (TN), 221 (FP) 124 (FN), 373 (TP)	314 (TN), 181 (FP) 218 (FN), 287 (TP)	318 (TN), 177 (FP) 135 (FN), 370 (TP)
<i>Our Model</i>		
328 (TN), 175 (FP) 154 (FN), 343 (TP)	366 (TN), 129 (FP) 244 (FN), 281 (TP)	387 (TN), 108 (FP) 178 (FN), 327 (TP)

Table 9: Confusion matrix comparing BERT (baseline) and our model at $T = 0$ on its respective balanced test set using the best-performing layers on the test set.

T = 0 (Question + Prompt)

Let the complex numbers z_1 and z_2 have corresponding points in the complex plane that are symmetric with respect to the real axis. Given that $z_1 = 1 + i$, find $z_1 z_2$. Let's think step by step:

T = 3 (Question + Prompt + 30% of the Generation)

Let the complex numbers z_1 and z_2 have corresponding points in the complex plane that are symmetric with respect to the real axis. Given that $z_1 = 1 + i$, find $z_1 z_2$. Let's think step by step: To find z_2 , we need to understand the symmetry of complex numbers with respect to the real axis.

When a complex number is symmetric with respect to the real axis, its imaginary part is negated.

Given $z_1 = 1 + i$, we can find z_2 by negating the imaginary part of z_1 .

S

T = 7 (Question + Prompt + 70% of the Generation)

Let the complex numbers z_1 and z_2 have corresponding points in the complex plane that are symmetric with respect to the real axis. Given that $z_1 = 1 + i$, find $z_1 z_2$. Let's think step by step: To find z_2 , we need to understand the symmetry of complex numbers with respect to the real axis.

When a complex number is symmetric with respect to the real axis, its imaginary part is negated.

Given $z_1 = 1 + i$, we can find z_2 by negating the imaginary part of z_1 .

So, $z_2 = 1 - i$.

Now, we need to find the product of z_1 and z_2 .

$z_1 z_2 = (1 + i)(1 - i)$

Using the difference of squares formula $(a + b)(a - b) = a^2 - b^2$, we can simplify the expression:

$z_1 z_2 = 1^2 - i^2$

Since $i^2 = -1$, we can substitute this value into the ex

Table 10: An Illustration of a sample from Cn-k12 dataset outlining how we construct the Prediction over Time dataset in our methodology.