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

Anum Afzal
Technical University of Munich
anum.afzal@tum.de

Florian Matthes
Technical University of Munich
matthes@tum.de

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

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]

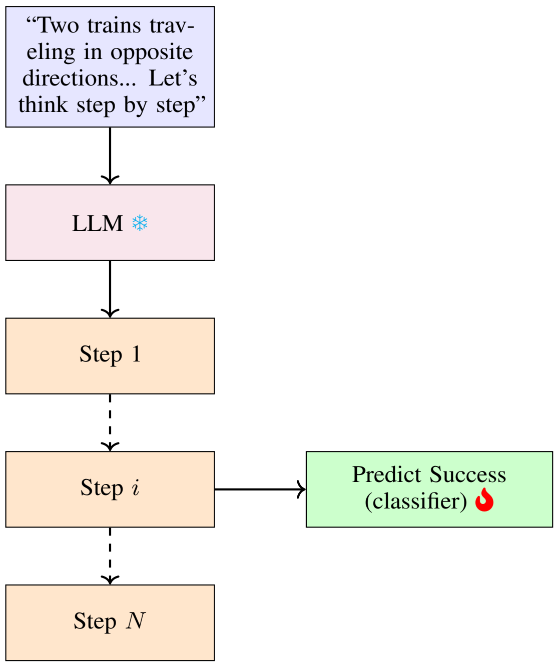
"Two trains traveling in opposite directions... Let's think step by step"
LLM ❄
Step 1
Step i
Step N
Predict Success (classifier) 🔥

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.

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), 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

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

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 (Belinkov, 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 effectiveness 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 $H = h_1, h_2, \ldots, h_L$ 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