PSEUDO CHAIN OF THOUGHT INDUCED FINE TUNING FOR LARGE LANGUAGE MODELS

Pranav Srinivasa (Corresponding Author)
Dept. Computer Science and Engineering
BMS Institute of Technology and Management
Bangalore, Karnataka - 560094
pranavsrinivas6@gmail.com

Ajith S
Assistant Professor
Dept. Computer Science and Engineering
BMS Institute of Technology and Management
Bangalore, Karnataka - 560094

Pratham P Punneshetty
Dept. Computer Science and Engineering
BMS Institute of Technology and Management
Bangalore, Karnataka - 560094

Pratham Gowda
Dept. Computer Science and Engineering
BMS Institute of Technology and Management
Bangalore, Karnataka - 560094

Niveditha Nayana P
Dept. Computer Science and Engineering
BMS Institute of Technology and Management
Bangalore, Karnataka - 560094

***Abstract* –** The current state of Large Language Models is filled with hallucinations and miscalculations, since the probabilities of the initial prompt and generations are not completely aligned with the probability distributions of the hypothetical right answer.

Therefore, a chain of thought process allows the model probability to align more closely to that of the expected output by reiterating over internal thoughts. The framework for inducing chain of thought to Large Language Model is to, first fine tune the main LLM on specific data required and further a smaller adapter model or also called as Tiny Thought Model is obtained from two-step model distillation process to get a model in the order of 100M-1B parameters that is focused on creating thoughts and questions based on inputs.

Finally, the Main LLM and the Tiny Thought Model are joined together via a router that decides if a thought is necessary to reiterate over the model answer, hence generating internal thought. A synthetic question answer pairs of the order of 5000-10000 QnA's are generated using an existing document such as a scientific or mathematical textbooks to train the Tiny Thought Model and also finetune the main LLM.

Further SAT Math and science datasets are used for generalized evaluations and test QnA's dataset for document specific evaluation. Measures such as Rouge, BLEU and Accuracy are going to be used to determine the quality of the answer. The base LLM with finetuning and base LLM without fine tuning are to be used as control measures. Models such as Llama 3.1 8B, Mistral 7B v0.3, Gemma 2 9B, Llama 3.2 10B will be used for comparisons.

1. INTRODUCTION

Large Language Models fail to address and provide correct answers for tasks that are not as simple as token generations since the model probability distribution does not align with the probability distribution of the answer. This can be fixed by providing more internal questions to allow the attention mechanism to align the model outputs to more closely match the output for the expected answers. A solution for this can be achieved by chain of thought prompting that requires a lot of manual labor for each prompt and increases the token size of input, or by the usage of large models that have internally adopted CoT such as the o1 and Claude models which have large param count, resource intensive and cost ineffective.

Finally, direct CoT finetuning of smaller LLMs serves as a solution for the problem but may cause catastrophically forgetting of the knowledge gained, hence the proposed framework adopts regular LLM fine tuning along with adapter LLM of the order of 100-1B parameters to be joined via a router to regulate the internal thoughts and serve as a safe induction of Pseudo Chain of Thought into smaller LLMs for specific data.

This method has the advantages of leveraging Chain of Thought for better outputs, with no knowledge loss and a low inferencing costs due to its smaller parameter count.

This allows the LLM to be finetuned with CoT of specific data according to the user and be run locally at lower costs as an alternative of using Retrieval Augmented Generations with larger LLMs.For the purposes of evaluation of the results from the proposed framework, the base LLM with regular finetuning and the base LLM without finetuning will be used as controls.

1.1. MOTIVATION

Large Language Models need to be smaller and easier to inference while having the ability to model the probability distribution of complex queries for effective usage for user specific data. We are trying to Improve the user experience with LLMs for company\user specific data, allow lower inferencing costs of LLMs and the ability for hosting locally, Leverage advanced mechanisms such as chain of thought at low inferencing and resource costs, making Large Language Models accessible and small, mitigate hallucinations of small LLMs by adopting Chain of thought. During the process of constructing this, it is crucial to optimize the data and architecture and training of these LLMs so that they can effectively model complex probability distributions without the need of massive amounts of resources.

This includes having a balance between reduction in model size and preservation of its capacity for understanding complex user queries. The project also seeks to ensure that these smaller LLMs remain aligned with the specific needs of users and organizations, providing accurate and context-aware outputs. When we integrate step-by-step reasoning into smaller language models, they can generate detailed, methodical responses while using fewer computational resources and reducing the likelihood of generating incorrect information, which frequently occurs in smaller models.

The proposed strategy will highlight how AI technology can be made easily available and customizable to a large scale of users. Implementing our proposed architecture, we can create efficient, compact language models that can run locally. This is a paradigm shift not only addressing key challenges such as cost and scalability but also empowers users to completely utilize the potential of AI-driven insights tailored to their unique contexts.

1.2. CONTRIBUTION

A comprehensive approach to induce chain of thought thinking into small scale foundational LLMs without interfering with its generational capabilities. The approach consists of fine-tuning the foundational LLMs of size ranging from 8B – 11B with appropriate data, here we aimed at creating math and python coding-oriented data from multiple datasets and created a standard template according to the Llama 3.1 8B model used and hence the foundational model is fine tuning using Low Rank Adaptation (LoRA).

Further, a dataset with the target text being Chain of Thought (CoT) instructions is created and a new foundational LLM of size ranging from 8B – 11B is finetuned using LoRA. This model undergoes a two-step knowledge distillation process to obtain a Tiny Thought Model (TTM) of size ranging from 1B – 3B which is capable of generating steps or guides for the main model to follow to arrive at the answer.

Hence by coupling the TTM with the finetuned foundational LLM via a Router which is a Network that predicts if a cycle of CoT needs to be induced, a complete small LLM is obtained that is finetuned on necessary data and can also perform Chain of Thought reasoning over multiple cycles without modifying the integrity of the LLMs generational capabilities.

Since fundamentally these are two LLMs one slightly bigger than the other, the LLMs along with the router can be run separately on different Cuda devices and enable horizontal scalability with very fast inferencing time.

2. OBJECTIVES

- To define the scope of the thought-specific model, Identify the domain, specify the tasks, and determine the representative dataset for specialization.
- To distill the large model into a compact thought-specific model, use knowledge distillation techniques to transfer reasoning capabilities into a smaller model (100M–1B).
- To implement an adapter framework for specialization, incorporate lightweight adapter layers into the large model and adapt it for the specific domain.
- To generate synthetic question-answer pairs, Leverage the large model to create diverse, high-quality Q&A datasets from document-specific data.
- To fine-tune the large model on the generated synthetic data, Train the large model on Q&A datasets to enhance domain-specific reasoning and task performance.
- To design a router neural network for reasoning depth, create a model to determine the optimal number of intermediate thoughts required for specific tasks.
- To train the router to connect models dynamically, Train the router to decide whether to use the large model or the thought-specific model for each query.
- To integrate the router with the fine-tuned and distilled models, Ensure seamless connection between the large model, the router, and the thought-specific model.
- To test the complete system for accuracy and efficiency, Validate the integrated system's outputs and measure its performance across diverse tasks.

- To iteratively refine the system based on performance feedback, continuously improve the models, router, and workflows to meet evolving requirements and optimize task outcomes.

3. METHODOLOGY

This is novel approach to induce a reasoning or thought process to any pre-trained large language model with small task specific datasets while also maintaining the knowledge base integrity of the pre-trained model.

This proposed methodology can quantitatively improve the accuracy within a large language model can respond with, since the probability density of the pre-trained model gets aligned to the hypothetical target probability through the reasoning steps involved. This alignment of probability will more closely mimic a problem-solving nature as opposed to the traditional predicting next token method and hence improve any base LLM for a certain task through reasoning. Furthermore, the qualitative nature of the LLMs output will parallel improve since the step-by-step view of a problem and the step-by-step solving of a problem inherently mimics human decision process and hence is proposed to have a greater resonance with the human chatbot interaction.

In the proposed approach the first step involves generating a synthetic dataset that only involves the question and a reasoning output for a specific question. This dataset is employed into training an 8-13B (Billion) parameter Large Language Model. The training process involves the use of the Low Rank Adaptation [11] (LoRA) method.

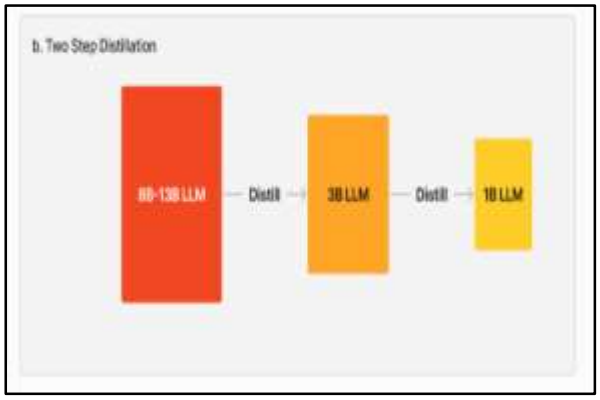
After the process of finetuning the Large Language Model to perform reasoning tasks, the knowledge is distilled using the Knowledge Distillation method employing KL-Divergence for Large Language Models [13]. This process involves a two-step distillation from the original 8B-13B model to a 3B model initially and then to a 1B model. This is done so that knowledge gained from the 8B-13B model can be easily passed down to a smaller model whereas a smaller model would be incapable of learning all the complex relations that can be modeled by the larger model. The resultant 1B LLM is modeled as the Tiny Thought Model or the TTM.

Furthermore, another base-LLM is chosen to be the main responder of the model. This Base-LLM is finetuned using the LoRA approach for any task specific use cases such as math, computer science, coding etc. A Router which is a feed forward network is trained based on the responses of the Base-LLM to decide when the optimal answer is reached compared to the question. Finally, the Tiny Thought Model and the Base-LLM are attached via the Router and hence Chain of thought is induced into the base-LLM.

During the process of a query asked by the user, the Tiny Thought Model first generates reasoning steps to be followed by the base model. This is then passed onto the base-llm and a response it given. Based on the response, the router either decides to stop the chain of thought process and serve the output or continues to prompt the Tiny Thought Model again to retrieve a fresh set of reasoning steps for the base-llm to follow.

4. MODEL ARCHITECTURE

The model architecture consists of a Tiny Thought Model (TTM) with 1B-100M parameters, a fine-tuned LLM with 7-13B parameters and a Router which is a feedforward MLP network. They are connected such that the overall

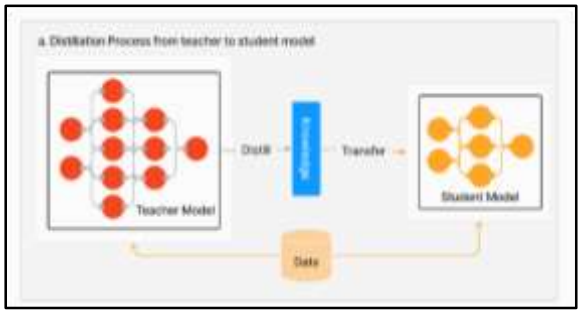


system mimics a Chain of thought structure with thoughts and reasoning cycles before providing the final answer, increasing the accuracy of alignment of model probabilities with real world probabilities.

This model architecture involves the depiction of the connections made to the finetuned large language models to be served. After the connection logic is implemented the model architecture can be served as a chatbot and overall model architecture can be thought of as on large language model that responds to user prompts through the validation of reasoning steps.

Fig. 4.1: Two Step Distillation Process.
Fig. 4.2: Distillation Process

Fig. 4.1 represents the process of obtaining the Tiny Thought Model for a Large Language Model of the order 8B-13B parameters. The process involves Two step distillation to avoid any knowledge that may occur in direct distillation from 8B-13B param model to a 100M-1B Tiny Thought Model. In this process of Two Step distillation involves first distillation from the Large Language model to an intermediate student model that is of the order 7B-13B params, further the intermediate student model is distilled again to obtain the Tiny Thought Model of 100M-1B parameter that is capable of closely matching the probability distribution of the parent model. The knowledge



distillation as shown in Fig. 4.2 loss function combines the supervised loss with the knowledge distillation loss in a weighted binary exponential format. Here's the mathematical formulation:

Let:

- $\mathcal{L}_{sup}$ note the supervised loss (e.g., cross-entropy loss between the student's predicted tokens and the target tokens).
- $\mathcal{L}_{KL}$ denote the Kullback-Leibler (KL) divergence between the logits of the student model and the teacher model.
- $\alpha \in [0,1]$ be the weighting factor controlling the trade-off between the two loss components.
- y represent the target tokens.
- p_s represent the probability distribution (logits) from the student model.
- p_t represent the probability distribution (logits) from the teacher model.

The loss components are expressed as:

Supervised Loss:

$$\mathcal{L}_{sup} = -\sum_{i} y_i \log(p_s(i)))$$

where iii indexes the output tokens.

KL Divergence Loss:

$$\mathcal{L}_{KL} = \sum_{i} p_t(i) log \frac{p_{t(i)}}{p_{s(i)}}$$

where p_t is the soft distribution from the teacher model and p_s is the soft distribution from the student model.

Combined Loss: The final loss combines these two components with exponential weighting:

$$\mathcal{L}_{final} = \alpha\mathcal{L}_{sup} + (1-\alpha)\mathcal{L}_{KL}$$

Fine-tuning large language models (LLMs) with LoRA involves adding low-rank updates to pretrained weights. Below is the concise format:

Base Model Forward Pass:

$$h = Wx$$

Where, $W \in \mathbb{R}^{d_{out} \times d_{in}}$ is the pretrained weight matrix, and $x \in \mathbb{R}^{d_{in}}$ is the input.

LoRA Weight Update:

$$W' = W + \Delta W, \qquad \Delta W = A.B$$

Where:

- $A \in \mathbb{R}^{d_{out} \times r}$ and $B \in \mathbb{R}^{r \times d_{in}}$ are learnable matrices, and r is the rank (typically small).

LoRA Forward Pass:

$$h = W'x = (W + AB)x = Wx + ABx$$

Scaled Low-Rank Update (optional):

$$\Delta W = \frac{\alpha}{r}AB$$

Where α is a scaling factor for stability.

Training Objective:

Optimize the loss with respect to A and B, while keeping W frozen:

$$\mathcal{L} = Loss(f(W'x), y)$$

Where y is the target, and f is the model function. This approach reduces the number of trainable parameters to $r \times (d_{out} + d_{in})$, enabling efficient fine-tuning.

Fig. 4.3 refers to the overall system architecture of the pseudo chain of thought fine-tuned model. Here, the base Large Language Model (LLM) is fine tuned to on the synthetic data from the document, further a trained Feed forward network is trained to invoke chain of thought optimal number of times before providing final improved response. This Feed Forward Network also called the Router is used to join the fine-tuned base LLM with the Tiny Thought Model obtained through two step distillation process referenced in Fig. 4.1 Therefore, after the system is constructed as shown in Fig. 4.3, based on an input from the user, the base LLM provides an output, based on which the router decides to invoke chain of thought, further the Tiny Thought Model creates an internal thought/question which is passed on to base LLM along with previous outputs and queries. This loop continues until the router network decides to exit reaching an optimal answer.

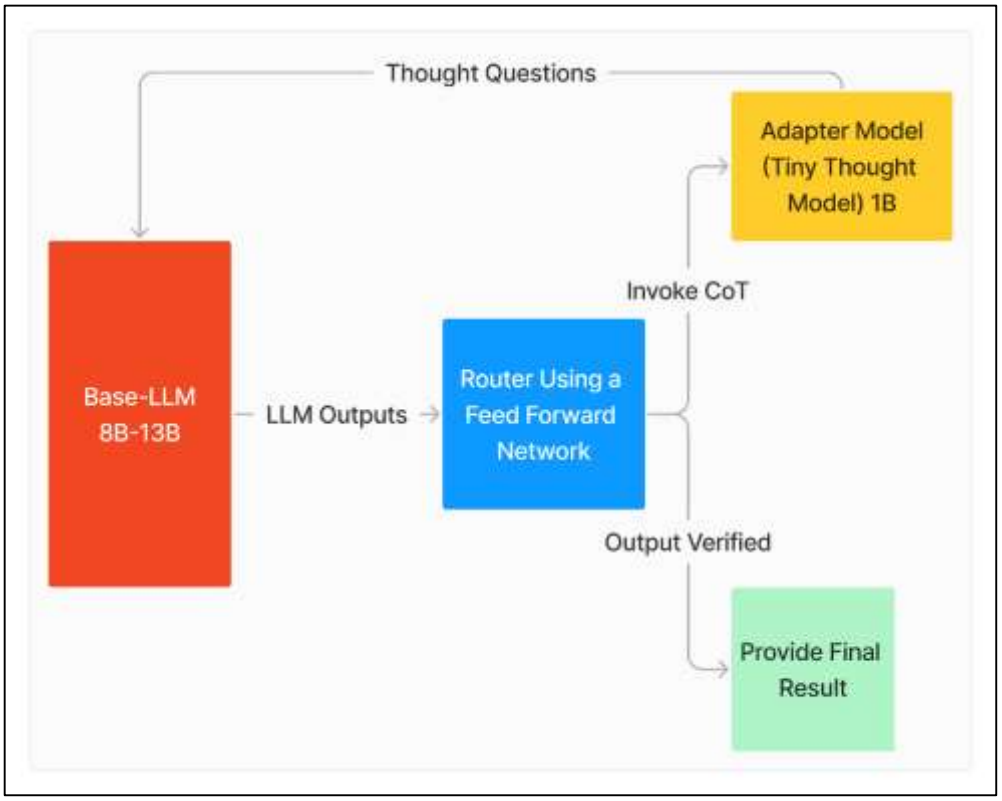


Fig. 4.3: Overview of model architecture.

5. IMPLEMENTATION

5.1. DATASET FORMATION

For the purposes of model distillation and creation of a Tiny Thought Model, a large dataset with reasoning steps for logical questions is to be generated. Here the target text for the model should not be the final answer and instead the intermediate reasoning steps. Hence several small datasets for math, science and computer science were collected and formatted with LLM tags, here we use Llama 3 suite of models hence the tags are <start_header_id>, <end_header_id>, <eot_id>, <end_message_id> and many more which signify certain format context to the LLM. After formatting all the datasets with system instructions, user responses and assistant responses, they are all merged and shuffled such that each subset contains the same percentage of each dataset's row. Further in the assistant responses the target text values are striped and hence it only remains with all the reasoning steps. For the purposes of model fine tuning similar steps are followed but the target text remains to be the final correct answer instead. The dataset is mainly an equally ratioed combination of multiple datasets consisting for complex mathematical, scientific questions and python coding along with reasoning steps.

5.2. HYPER PARAMETRIC SETTINGS

5.2.1. LoRA Fine Tuning of Base LLM:

For faster training and resource efficiency a 4-bit quantized Llama 3.1 8B model is opted to be the base LLM. With the help of Unsloth fine tuning a QLoRA finetuning framework was developed for the Base LLM with mathematics expertise dataset. Here, the Query, Key, Value Matrices along with Up and Down projections were enabled for the fine-tuning process. It utilized a binary float 16 datatype for weights with a linear learning rate scheduler with the target learning rate being $2 * 10^{-4}$. The rank was set to 32 and alpha to 64 with the batch size being 8 per device. Due to resource constraints the maximum step size was set to 300.

5.2.2 Knowledge Distillation for TTM:

For the purposes of knowledge distillation, the process of normal finetuning took place for a Llama 3.1 8B with a reasoning specialized dataset and hence was used in finally distilling to a Llama 3.2 1B model. To implement Knowledge Distillation a standard Supervised Fine-Tuning Trainer was implemented with the compute class being overridden to implement the knowledge distillation loss as mentioned above in section 4.
Here, only the Query and Key matrices were enabled for distillation and a batch size of 2 with a maximum training step size of 200 was implemented. Learning rate schedulers were similar to the parametric settings in LoRA finetuning process mentioned in section 5.2.1.

5.3. INFERENCE AND EVALUATION

For the purposes of evaluation, the Base LLM was set to a greedy decode method, with a top K being 100 and top P being 0.95, due to resource constraints. For inference purposes the Base LLM was set to a beam search decode method with the number of beams being 2. The Temperature for the base LLM was set to 0.6 with a max new token of 2048, during both evaluations and inference to promote more accurate results.
The Tiny Thought Model was set to a beam search decode method with the number of beams being 2 and a max new token of 512 during both evaluations and inference.
For evaluation and training both the models were prefixed with certain system instructions and a 1 shot example for the output format.
All the above settings were constant for all the comparison models used to establish a control in the environment.

6. RESULTS AND DISCUSSION

Finally, after the process of two step knowledge distillation using a combination of cross entropy loss and reverse kl divergence loss, we obtain a Tiny Thought Model of size 1B reduced from 8B. The Base LLM can be unaltered but for task specific applications such as math and reasoning fine tuning on an appropriately large dataset for the task improves the overall answering capacity and internal knowledge in the probability distribution of the LLM.

6.1. METRIC DESCRIPTION.

Finally, after the process of two step knowledge distillation using a combination of cross entropy loss and reverse kl divergence loss, we obtain a Tiny Thought Model of size 1B reduced from 8B.
The Base LLM can be unaltered but for task specific applications such as math and reasoning fine tuning on an appropriately large dataset for the task improves the overall answering capacity and internal knowledge in the probability distribution of the LLM.

After Combining both these models as mentioned in the proposed methodology, evaluation on standard datasets such as Massive Multitask Language Understanding
Meaning (MMLU), with the college math and college computer science split, is carried out with metrics being

Accuracy, bilingual evaluation understudy (BLEU), Recall-Oriented Understudy for Gisting Evaluation (ROUGE).

The Bilingual Evaluation Understudy (BLEU) has gained a lot of popularity as a metric used in evaluating machine translators as well as text generation models. It calculates n-gram matches with various reference texts and implements a precision measure at the same time taking into account a brevity penalty for short generated outputs.

The Recall-Oriented Understudy for Gisting Evaluation (ROUGE) on the other hand is used for the assessment of text summarization and generation on a recall basis with n-grams overlapping. Some of the families which ROUGE is composed of are ROUGE-N, ROUGE-L, and ROUGE-W which check for n, longest and weighted sequences respectively

6.2. COMPARISON OF MODELS

S.NO.	*MODELS*	*MMLU COLLEGE MATHS*			*MMLU COLLEGE CS*		
		ACCURACY %	*BLEU*	*ROUGE*	*ACCURACY %*	*BLEU*	*ROUGE*
1	Llama 3.1 8B	22	0.1300	0.0500	49	0.1700	0.1900
2	Llama 3.1 8B CoT	32	**0.5300**	**0.5000**	**50**	**0.6800**	**0.6800**
3	Llama 3 8b math finetuned	26	0.1298	0.0313	22	0.0590	0.0201
4	Llama 3 8b math finetuned with CoT	**37**	0.1158	0.0137	33	0.0652	0.0233
5	Mistral 7b v0.3	27	0.1174	0.0139	30	0.0656	0.0203
6	Gemma 2 9B	29	0.1134	0.0129	39	0.0615	0.0190

Table 6.1: Quantitative results of evaluation over selected benchmark

The table 6.1 depicts the values achieved by each of the selected models for the benchmark of MMLU Mathematics and MMLU Computer Science, each being at college level. The Models chosen for the purpose of comparison involves the base models (Llama 3 8B) to establish a control and some commercially available models of comparable parameter counts such as the Mistral 7B v0.3, Gemma 2 9B.

The metrics used here are Accuracy percentage, BLEU and ROUGE L1. The highlighted values at each of the metric section for each of the selected benchmark depicts the highest score achieved in the respective benchmark for the respective metric.

As observed the highlighted values are skewed towards models with an ability to have a chain of thought. Hence a

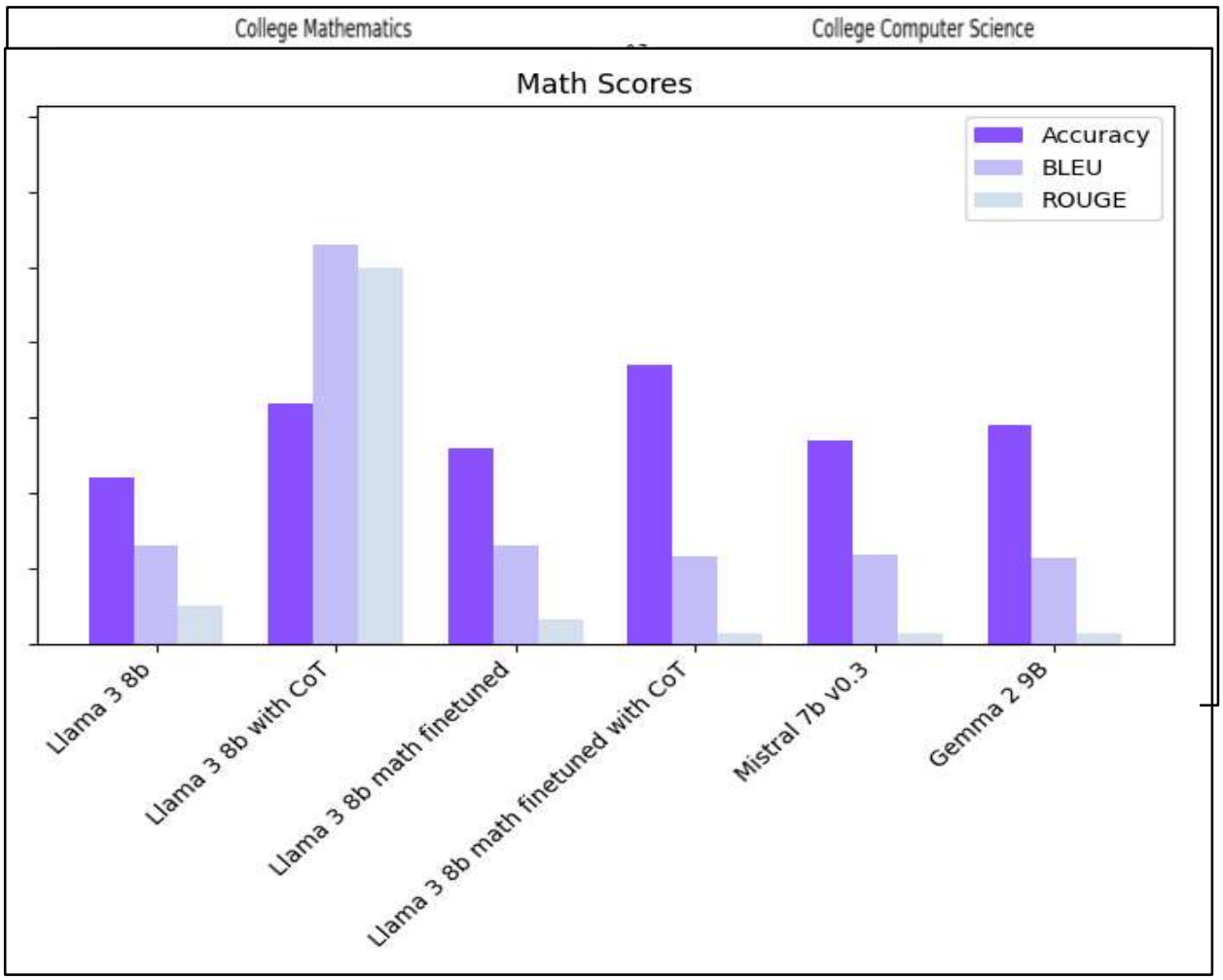


Fig. 6.2: Comparison of College Computer Science Scores with other models

Rudimentary analysis results in association of better performance with Chain of thought. Hence with the proposed methodology the models with induced chain of thought have achieved higher scores as compared to the control base large language models while also achieving better performance as compared to other open-source models of comparable parameter count such as Mistral 7b and Gemma 2 9b.

However, the increase in performance comes with an additional parameter count of 1 billion parameters to account for the Tiny Thought Model. But even with the higher parameter count the accuracy achieved is higher even with similar parameter count models such as the Gemma 2 9b.

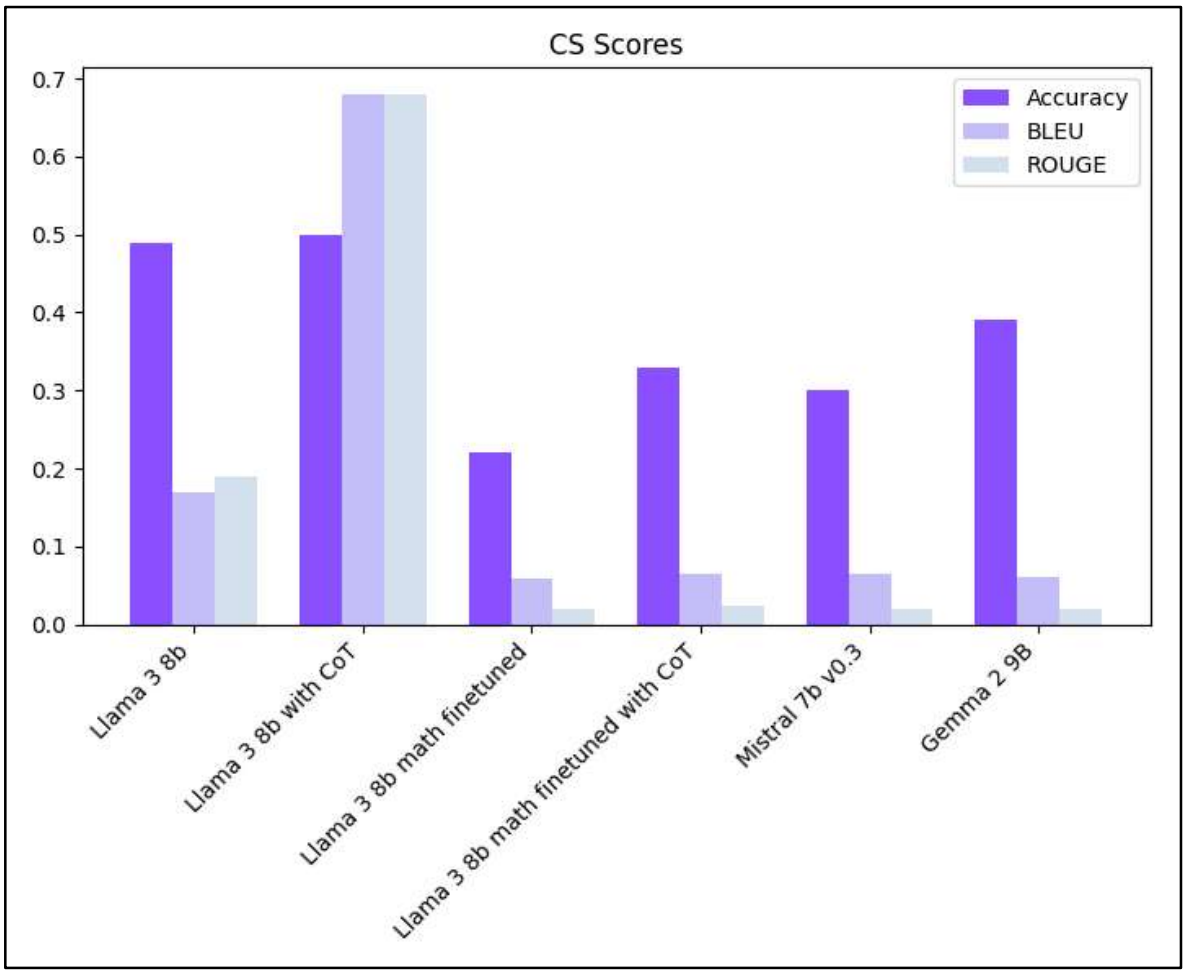


Fig. 6.3: Comparison of College Mathematics Scores with other models

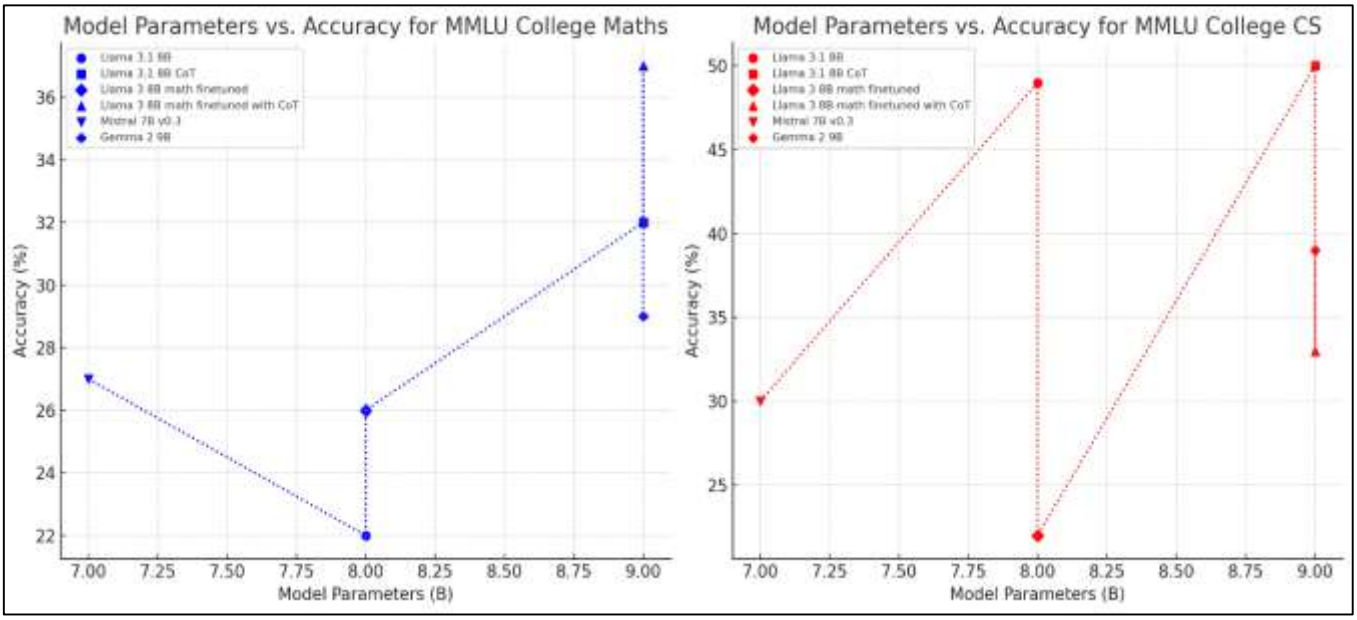


Fig. 6.4: Model Parameter vs Accuracy for each evaluation benchmark.

The Fig. 6.1 shows the comparison between the base LLM and CoT fine-tuned LLM. This shows a significant improvement of the CoT induced LLM in math with a 10% increase in accuracy rates and significantly higher BLEU and ROUGE scores which indicate better explainability of LLM for the answer. This shows that even without any Knowledge finetuning of base LLM, and only by inducing Chain of Thought TTM, the model performance is perceivably better.

The Fig. 6.2 Shows the comparison of the MMLU College level computer science evaluations of the proposed Pseudo CoT induced model with the publicly available models while using the base model as comparison. Here due to lack of fine-tuning knowledge of the distribution of data, the non-fine-tuned, Chain of Thought induced model performs the best while also performing better than all the other models such as Gemma 2 9B, Mistral 7B v0.3. The Fig. 6.3 also shows similar results but comparison of MMLU Mathematics yields the Fine-Tuned Pseudo CoT model as the victor due to training on similar distribution data. Hence this shows the models capability on task specific operations which require reasoning.

The Fig. 6.4. depicts the comparison between the model size over the model accuracy achieved on each of the bench marks. The models with CoT induced finetuning have been increased by 1B parameters accounting for the Tiny Thought Model. Even with this slight initial increase, we can deduce that there exists a large improvement over the accuracy compared to the base model aswell as the finetuned model, hence making the trade off between the between increased model size to accuracy is validated and justified. Further compared to models of similar parameters size and higher parameter size compared to the base model chosen, it is observed to have a substantial improvement, hence validating the additional resources to induce chain of thought to a smaller model as opposed to using a commercially available larger model.

Therefore, as seen by the evaluation, through the process of chain of thought induction as proposed in the methodology the Large Language Model clearly outperforms the Base-LLM itself as well as improve over certain models available apart from the base model with equal or slightly greater parameter. Hence for the use for real world applications, this process can be applied to improve certain areas of knowledge and responses such as mathematics and computer science as shown in the evaluations.

However since the training process was done for a relatively small amount of time, the chain of thought induced models are not at peak performance, and hence the evaluation shows better performance increase towards mathematics and numerical data over textual data.

7. CONCLUSION

7.1. SUMMARY

The proposed Pseudo Chain of Thought Induced Finetuning of model is a significant advancement in improving the reasoning and task-specific capabilities of large language models (LLMs). Through the two-step knowledge distillation process, the Tiny Thought Model (TTM) is reduced from 8B to 1B parameters. Further, fine-tuning the Base LLM on task-specific datasets, particularly in mathematics and reasoning, has significantly improved the model's answering capability.

Evaluation using the Massive Multitask Language Understanding (MMLU) dataset revealed key findings In mathematics performance a 10% increase in accuracy compared to the Base LLM is observed, with higher BLEU and ROUGE scores, highlights improved explainability and reasoning, Computer Science Performance without knowledge fine-tuning, the CoT-induced model outperforms all publicly available alternatives such as Gemma 2 9B and Mistral 7B v0.3.

These results underscore the efficacy of combining the distilled TTM and a fine-tuned Base LLM for achieving superior performance in both general and task-specific reasoning challenges but there exists a lot of scope for improvement of the current results which let the model outperform higher parameterized models as well.

7.2. FUTURE WORK

These improvements mainly are, scaling model size of the TTM and the Base LLM have substantial performance gains. Exploring TTM sizes of 3B or 4B could balance efficiency with reasoning depth, while scaling the Base LLM to 13B or larger.

Increasing Training Time for the distillation and fine-tuning process, especially with larger datasets, can improve the alignment of the TTM's reasoning capabilities and the Base LLM's knowledge.

Dynamic Routing Optimization between the TTM and the Base LLM to better identify tasks requiring detailed reasoning could speed up performance. Adaptive mechanisms could be developed for selecting reasoning depth based on task complexity. Specialized CoT Techniques exploring domain-specific Chain of Thought patterns.

In conclusion the proposed model marks a substantial leap in reasoning and task-specific performance, paving the way for future advancements that scale both the architecture and its training paradigm.

7.3. QUALITATIVE ANALYSIS.

Fig. 7.1: Example Output of CoT induced LLM.

From the above figure, fig. 7.1, we can an observe how a model, that has been induced with Chain of Thought through processes that have been proposed in the section 4 of this paper, can be deployed locally. This can be considered a reference user interface that shows the output of the tiny thought model and the main model. When prompted with a physics question. We can observe that Tiny Thought model provides a list of steps to follow that might lead to the correct answer. Finally, the main model interprets the steps and hence provides a comprehensive answer.

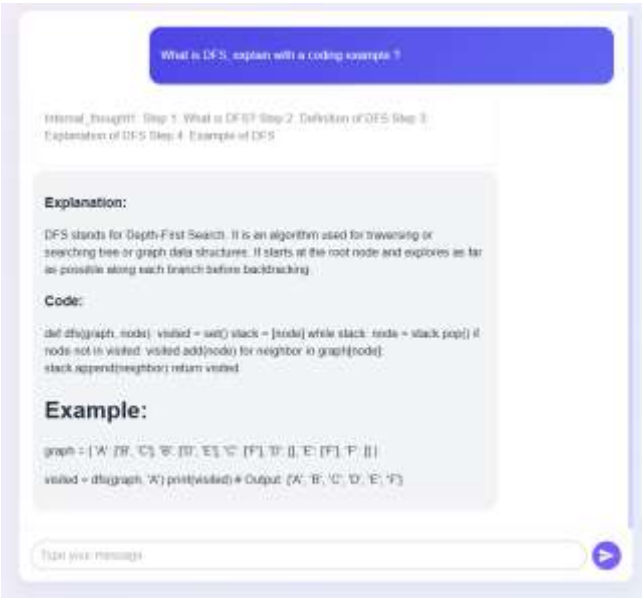


Fig. 7.2: Sample Output of CoT Induced LLM

As observed the internal thoughts that are produced by the Tiny Thought Model, provides a step-by-step guide for the main model to understand and hence perform task and provide outputs beyond its normal capabilities

7.4. REAL WORLD APPLICATIONS.

The combination of the two Large Language Models is negligible since the distilled Tiny Thought Model only constitutes to 1.2GB of additional memory in comparison to the ~ 6-7GB of memory utilized by the Main model.

Hence due to the relatively small size and performance boost the combined model architecture can be used for various small to medium tasks, some of them being, Code base reviewing, Customer Care Agents, Customer Relations handling and many more.

The small size and parallelizable properties of the proposed architecture allows the model to be self-hosted at minimal cost for small and medium tasks that are not at the highest of priorities and provide great cost-effective solutions compared to the utilizing Large Foundational Models of the order 50-100B for tasks that do not require large models and instead requires reasoning abilities. The integration of the Pseudo Chain of Thought (Pseudo-CoT) technique into Large Language Models (LLMs) offers several pragmatic

real-world uses for sectors that require improved reasoning, efficiency, and cost savings. The combination of a compressed Tiny Thought Model (TTM) and Base LLM fine-tuned is intended to improve computational efficiency with high task completion accuracy rates. A Subset of such usage is illustrated below:

- **Automated Review and Debugging of Code**
    - Pseudo-CoT framework's ability to generate step-by-step reasoning makes it extremely valuable in coding and debugging.
    - The TTM can identify logical errors in pieces of code through breaking the problem down into fragments.
    - The Base LLM can then examine the suggested solution and check for correctness against best practice and coding guidelines.
    - The system can be implemented as virtual assistants in IDEs for live coding optimization recommendations.
- **Virtual Assistants and Customer Support**
    - The ability of the system to minimize hallucinations and facilitate rational coherence is applicable to the needs of virtual customer support assistants.
    - The TTM is able to parse customer questions and create a rational thought process to solve intricate customer queries.
    - The Base LLM helps ensure that answers are contextually relevant and policy compliant for companies.
    - This technique greatly enhances accuracy of responses and minimizes human agent dependency, decreasing operational expenditures.
- **Educational Tutoring Systems**
    - Using its improved reasonability, the Pseudo-CoT model has the potential to transform personalized learning platforms by rendering elaborate explanations for intricate ideas.The TTM produces intermediate reasonability steps in order to answer questions, and therefore is ideally suited for usage in mathematics and science education.
    - The Base LLM provides reliable final answers, minimizing the possibility of presenting false information to students. Adaptive learning technology may utilize this model to modify levels of difficulty dynamically in accordance with students.
- **Medical Diagnosis Support**
    - Pseudo Chain of Thought step-by-step reasoning style may be applied in medical diagnosis systems through AI. The TTM helps break down symptoms into potential underlying conditions.
    - The Base LLM, being trained on healthcare data, is able to provide a probabilistic diagnosis in such a manner that practitioners receive structured guidance. It is particularly useful in telemedicine applications where live AI assistance required.
- **Financial and Legal Document Analysis**
    - Legal and finance sectors rely on the extensive explanation for contract analysis, compliances, and fraud detection.
    - The TTM is capable of extracting and generating step-by-step logical interpretations of legal document clauses.
    - The Base LLM verifies compliance with regulatory rules for contractual correctness. The model may be integrated into automated compliance applications used by banks and law firms.

REFERENCES

[1] Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.
[2] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, *35*, 24824-24837.
[3] Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., ... & Le, Q. V. (2021). Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.
[4] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., ... & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

[5] Saparov, A., & He, H. (2022). Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240*.
[6] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, *33*, 9459-9474.
[7] Perez, E., Lewis, P., Yih, W. T., Cho, K., & Kiela, D. (2020). Unsupervised question decomposition for question answering. *arXiv preprint arXiv:2002.09758*.
[8] Hoffman, M. D., Phan, D., Dohan, D., Douglas, S., Le, T. A., Parisi, A., ... & A Saurous, R. (2024). Training chain-of-thought via latent-variable inference. *Advances in Neural Information Processing Systems*, *36*.
[9] Kim, S., Joo, S. J., Kim, D., Jang, J., Ye, S., Shin, J., & Seo, M. (2023). The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.14045*.
[10] Mitra, C., Huang, B., Darrell, T., & Herzig, R. (2024). Compositional chain-of-thought prompting for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14420-14431).
[11] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
[12] Li, Y., Yu, Y., Liang, C., He, P., Karampatziakis, N., Chen, W., & Zhao, T. (2023). Loftq: Lora-fine-tuning-aware quantization for large language models. *arXiv preprint arXiv:2310.08659*.
[13] Gu, Y., Dong, L., Wei, F., & Huang, M. (2023). Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.
[14] Lin, J., Tang, J., Tang, H., Yang, S., Chen, W. M., Wang, W. C., ... & Han, S. (2024). AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. *Proceedings of Machine Learning and Systems*, *6*, 87-100.
[15] Zhang, Z., Zhang, A., Li, M., & Smola, A. (2022). Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.
[16] Lyu, Q., Havaldar, S., Stein, A., Zhang, L., Rao, D., Wong, E., ... & Callison-Burch, C. (2023). Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.

[17] Madaan, A., & Yazdanbakhsh, A. (2022). Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*.
[18] Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., ... & Wei, J. (2022). Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.
[19] Diao, S., Wang, P., Lin, Y., Pan, R., Liu, X., & Zhang, T. (2023). Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*.
[20] Wang, B., Deng, X., & Sun, H. (2022). Iteratively prompt pre-trained language models for chain of thought. *arXiv preprint arXiv:2203.08383*.
[21] Zhang, Z., Yao, Y., Zhang, A., Tang, X., Ma, X., He, Z., ... & Zhao, H. (2023). Igniting Language Intelligence: The Hitchhiker's Guide From Chain-of-Thought Reasoning to Language Agents. *arXiv preprint arXiv:2311.11797*.
[22] Chia, Y. K., Chen, G., Tuan, L. A., Poria, S., & Bing, L. (2023). Contrastive chain-of-thought prompting. *arXiv preprint arXiv:2311.09277*.
[23] Yao, Yao, Zuchao Li, and Hai Zhao. "Beyond Chain-of-Thought, Effective Graph-of-Thought Reasoning in Language Models." *arXiv preprint arXiv:2305.16582* (2023).
[24] Lanham, T., Chen, A., Radhakrishnan, A., Steiner, B., Denison, C., Hernandez, D., ... & Perez, E. (2023). Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.
[25] Zhang, X., & Ding, D. (2024). Supervised chain of thought. *arXiv preprint arXiv:2410.14198*.