# Rational Metareasoning for Large Language Models

**C. Nicolò De Sabbata**[1,2] * **Theodore R. Sumers**[3] **Thomas L. Griffiths**[1]

[1]Princeton University [2]EPFL [3]Anthropic

## Abstract

Reasoning has emerged as a core technique for improving large language model (LLM) performance across various tasks by using additional inference-time compute. However, as LLMs scale in both size and usage, inference costs are becoming increasingly burdensome. How, then, might we optimize the cost-performance tradeoff of reasoning? This work introduces a novel approach based on computational models of metareasoning used in cognitive science, training LLMs to selectively use intermediate reasoning steps only when necessary. We first develop a reward function that incorporates the Value of Computation by penalizing unnecessary reasoning, then use this reward function with Expert Iteration to train the LLM. Compared to few-shot chain-of-thought prompting, our approach significantly reduces inference costs (38% fewer tokens generated on average) without sacrificing task performance across diverse datasets.

## 1 Introduction

Large language models (LLMs) rely on substantial computational power to handle complex problems [1, 2, 3]. While initial studies mostly focused on the cost of training [4], LLMs' widespread deployment has made inference-time costs an increasingly important factor. Model compression techniques such as quantization, pruning, and knowledge distillation can lower post-training costs [5]. However, there is a fundamental tension between inference cost and task performance: while many of these methods reduce costs at the expense of performance, others, such as chain-of-thought reasoning (CoT) [6, 7], do the opposite, raising inference costs to enhance task performance. Crucially, however, none of the previous approaches are *adaptive*: model compression modifications and existing CoT methods raise or lower the inference cost on *all* queries, regardless of task complexity.

In stark contrast to this static tradeoff, humans are able to adaptively allocate computational resources based on task difficulty [8, 9, 10]. In this work, we draw inspiration from *rational metareasoning* – literally, reasoning about reasoning – a concept originally from the artificial intelligence literature that has been used to explain how humans adaptively manage computational resources [10, 11, 12, 13]. Building on this, we develop a novel reward function based on the Value of Computation (VOC, [11]), which formalizes the trade-off between inference cost and task performance. We adopt an iterative reinforcement learning process inspired by the Expert Iteration algorithm [14]. In each iteration, we generate multiple rationales for each question. These rationales are ranked using the reward function, and the dataset is filtered to retain only the best rationale for each question. The model is then fine-tuned using this filtered dataset. This method was strongly inspired by STaR [15], which bootstraps reasoning ability through a self-improvement loop. However, unlike STaR, which filters generated examples based solely on whether their final answer matches the target, our method evaluates reasoning not just on the correctness of the final answer, but also considers the cost and utility of the reasoning process itself. We evaluated the effectiveness of our solution across a diverse set of tasks, from science knowledge (ARC [16]) to commonsense reasoning (CommonsenseQA [17]),

*Work done as visiting student researcher at Princeton University

NeurIPS 2024 Workshop on Behavioral Machine Learning.

mathematical problem solving (GSM8K [18]), and logical deductive reasoning (ProofWriter [19]). Additionally, we assess the out-of-domain generalization on MMLU [20], a multitask benchmark. Our approach achieves a substantial reduction in generated tokens (38% compared to few-shot prompting and 22% compared to STaR [15]) while retaining comparable performance. Related works [21, 22, 23, 24, 15, 25, 26] have expanded upon CoT, further enhancing performance across various scenarios. However, these approaches do not improve efficiency – in fact, they often generate even longer sequences. Finally, adaptive computation methods often involve training multiple models or altering architectures [27, 28, 29, 30, 31], whereas our approach modifies only the fine-tuning process of pretrained LLMs.

## 2 Rational Metareasoning

Humans have limited time and cognitive resources [32, 33]. We face diverse challenges requiring different approaches: avoiding a sudden obstacle while driving needs quick, intuitive thinking, while selecting a retirement investment strategy requires slow, deliberate reasoning [8]. Rational metareasoning [11] suggests agents should adapt their reasoning based on the problem at hand. Intuitively, while reasoning solves a problem, metareasoning solves the problem of *how* to solve a problem: deciding which computations to perform while problem-solving. In formal terms, rational metareasoning can be distilled into the problem of calculating the value of computation (VOC) [11] for each potential computation ($c$). The VOC balances the benefit of computation $c$ (characterized by the expected increase in utility) against its cost (usually time or energy).

To formalize this, agents are assumed to have some internal belief state $b \in \mathcal{B}$, which determines their expectation about the value of each action $a \in \mathcal{A}$: $\mathbb{E}[U(a)|b]$. A rational agent would simply choose the highest-value action: $a^* = \arg\max_{a \in \mathcal{A}} [U(a)|b]$. In contrast, a meta-rational agent can perform computation to change their belief state before choosing an action. Each computation $c \in \mathcal{C}$ has an associated cost (cost($c$)), but updates the agent belief to $b'$ with probability $P(b'|c)$. This, in turn, affects their beliefs about the value of actions. Formally, then, the VOC quantifies the value of performing computation $c$ given a starting belief state $b$,

$$VOC(c, b) = \mathbb{E}_{P(b'|c)}[\max_{a'} \mathbb{E}[U(a')|b'] - \max_{a} \mathbb{E}[U(a)|b]] - \text{cost}(c). \tag{1}$$

Thus, a meta-rational agent should pursue the computation $c^*$ with the highest VOC: $c^* = \arg\max_{c \in \mathcal{C}} VOC(c, b)$. If no computation has positive VOC, the agent should stop thinking and instead act in the world. Rational meta-reasoning has been used to explain how humans allocate cognitive resources in various tasks [10, 12].

## 3 Introducing Rational Metareasoning into Large Language Models

To achieve an optimal balance between performance and efficiency, our approach introduces a new VOC-inspired reward function (Eq. 3) into an Expert Iteration training loop [14, 15], fine-tuning a LLM to produce rationales adaptively depending on task difficulty.

**Reward modeling** We define the reward of a chain of thought as the difference between its utility and its cost,

$$\mathcal{R}_\pi(x, y, z) = \mathcal{U}_\pi(z|x, y) - \mathcal{C}(z) \qquad \mathcal{U}_\pi(z|x, y) = \log \pi_\theta(y|z, x) - \log \pi_\theta(y|x) \tag{2}$$

where $x$ denotes the input for the task, $z$ represents the intermediate chain of thought, and $y$ is the target solution. The utility of the chain of thought is represented by $\mathcal{U}_\pi(z|x, y)$, and the cost of the intermediate computations is denoted by $\mathcal{C}(z)$. In the context of LLMs, utility quantifies the increase in the likelihood of generating the target sequence $y$ when the chain of thought $z$ is added to the input $x$, under the policy $\pi$. Specifically, $\pi_\theta(y|z, x)$ indicates the probability of generating the target sequence $y$ given both the chain of thought $z$ and the input $x$, while $\pi_\theta(y|x)$ denotes the probability of generating $y$ with only the input $x$. The cost, on the other hand, is directly proportional to the number of tokens in the chain of thought $l(z)$: $\mathcal{C}(z) = \gamma \cdot \log l(z)$. The hyperparameter $\gamma$ scales the cost and utility to the same magnitude. A key benefit of this reward function is that it is parameterized by the same weights $\theta$ as the generative policy $\pi_\theta$, eliminating the need for an external reward model. This allows direct estimation of a rationale's utility using the policy itself.

**Rationale Generation.** We start with a pretrained language model $\pi_\theta$ and an initial dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$ of problems $x$ and their correct answers $y$. To train the model to generate rationales,

we adopt a few-shot prompting method by prepending a small set of examples $\mathcal{P}$, each containing intermediate rationales $z$, to each $x$ in $\mathcal{D}$. For each task $\tau_i = (x_i, y_i)$, we produce $K$ rationales: $\hat{\tau}_i = \{(x_i, z_{k,i}, y_i)\}_{k=1}^{K}$. If none of the $K$ rationales for $\tau_i$ is correct, we discard all samples for that task. To reduce the likelihood of incorrect rationales, we use the rationalization method from STaR [15], where a new rationale is generated using the correct answer if the initial attempts fail. Each rationale is then evaluated using the Rational Metareasoning reward function.

**Metareasoning Training.** We demonstrate the effectiveness of our reward using a variation of the Expert Iteration (EI) algorithm [14]. EI is known for its sample efficiency and strong performance on reasoning tasks [34, 15]. As an example of an online reinforcement learning (RL) algorithm, EI involves both exploration and policy improvement phases, with the policy $\pi_\theta$ being updated using data from the exploration phase. The training process is described in Algorithm 1.

**Algorithm 1** Rational Metareasoning Training

**Input** $\pi$: a pretrained LLM; dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$
**for** $n$ **in** $1...N$ **do** ▷ Iterations
  $\mathcal{D}_n \leftarrow \mathcal{D}$ ▷ Sample batch from dataset
  **for** $k$ **in** $1...K$ **do** ▷ Perform rationale generation
    $(z_{i,k}, y_{i,k}) \leftarrow \pi_{n-1}(x_i) \quad \forall i \in [1, D_i]$
    $(z_{i,k}, y_{i,k}) \leftarrow \pi_{n-1}(\text{add_hint}(x_i)) \quad \forall i \in [1, D_i] \wedge y_{i,k} \neq y_i$ ▷ Compute rationalization
  **end for**
  $r_{i,k} \leftarrow \mathcal{R}(x_i, y_i, z_{i,k}) \quad \forall i, k (i \in [1, D_i] \wedge y_{i,k} = y_i)$ ▷ Compute reward for each rationale
  $\hat{z}_i \leftarrow \arg\max_k \{\mathcal{R}_\pi(x, z_k, y)\}_{k=1}^{K}$ ▷ Select best rationale for each task $i$
  $\mathcal{D}_n^* \leftarrow \{(x_i, \hat{z}_i, y_i) \in [1, D_i]\}$ ▷ Create the optimal dataset
  $\pi_n \leftarrow \text{train}(\pi, \mathcal{D}_n^*)$ ▷ Finetune the original model on the optimal solutions
**end for**

Initially, in the exploration phase, we approximate the optimal policy $\hat{\pi}^*$ by using rejection sampling on our student policy $\pi_\theta$. After generating $K$ intermediate rationales $z_1, \ldots, z_K$ for a given question $x$, we evaluate them using our reward function $\mathcal{R}_\pi$ (the rationale generation process described in 3). We then construct $\mathcal{D}_1^* = \{(x_i, z_i, y_i)\}_{i=1}^{N}$ by selecting the best rationale for each task $i$: $z_i = \arg\max_k \{\mathcal{R}_\pi(x, z_k, y)\}_{k=1}^{K}$. These rollouts are then distilled into a policy $\pi_1$ using standard cross-entropy loss. This process can be iteratively repeated to refine the policy $\pi_n$ on the dataset $\mathcal{D}_n^*$.

Instead of using the entire training dataset at each iteration, as standard EI algorithms do [14, 35], we start with a batch of $T$ steps at the first iteration and increase the number of fine-tuning training steps by $T$ at each subsequent iteration (similar to STaR [15]). This approach allows the model to encounter new examples gradually, resulting in slower training initially, which ultimately enhances model performance.

## 4 Experiments

**Datasets.** We constructed our training set by combining the training sets from these datasets into one dataset $\mathcal{D}$ and then evaluated the model on all corresponding test sets $\mathcal{T}$. We used the following datasets: **ARC** [16], which includes natural science questions to assess scientific understanding; **CommonsenseQA** [17], focused on everyday reasoning; **GSM8K** [18], with diverse math word problems for arithmetic skills; and **ProofWriter** [19], which tests logical deductive reasoning by evaluating conclusions from given premises. To balance the datasets and manage costs, we randomly sampled 1,024 entries from each training set. We conducted further testing using the first 100 samples from each of the 57 subjects in the **MMLU** benchmark[20], which includes multiple-choice questions on various topics, to evaluate the model's generalization ability.

**Baselines.** We illustrate the advantages of our model by comparing its performance to two types of prompting strategies: **direct**, where the model is required to provide an immediate answer, and **Chain of Thought (CoT)**, where the model is encouraged to reason through the problem step-by-step before arriving at a solution. Since we are using pretrained models (which are not specifically trained for instruction following), we provide five few-shot examples for each task. These examples are carefully chosen to ensure that the length of the rationale matches the perceived difficulty of the question. In addition to these prompting methods, we adopt a finetuning baseline, comparing our method to **STaR** (Self-Taught Reasoner [15]), which also uses the Expert Iteration algorithm.

**Training Details.** For our experiments, we use Meta Llama-3-8B [36] as the pretrained base model. To generate rationales, we sample $K = 5$ sequences for each question, using a temperature $t$ of 0.5 and a $top_p$ value of 0.9. For the reward, we set $\gamma = 0.1$ to balance utility and cost on the same scale. For each iteration $n$, we sample a dataset $\mathcal{D}_n$ of size 512 from the union of four training datasets $\mathcal{D}$ described in 4. In the self-supervised fine-tuning step, we use a batch size of 16 and a learning rate of 1e-6. Finally, we evaluate all models using greedy decoding to ensure consistent and deterministic output generation. We use pattern matching techniques to extract the answers; an exact match between the generated answer and the ground truth is considered correct.

**Results.** As shown in Figure 1, even in datasets where the Chain of Thought method yields marginal performance gains, the length of the generated rationales remains substantial. Conversely, with our method, not only is the overall length reduced, but the difference in the average rationale length between the two dataset groups becomes more pronounced. This indicates that the model has learned to more effectively distinguish when detailed reasoning is necessary and when a shorter response is sufficient. Our training method shows that the LLM maintains accuracy comparable to traditional few-shot prompting, with only minor variations depending on the dataset, while achieving significant efficiency gains. Specifically, our approach reduces input tokens by an average of 94% and output tokens by 38%, without compromising performance. Compared to the STaR method, our approach achieves similar performance levels while generating 22% fewer tokens on average.

| **Method** | **Accuracy (%)** $\uparrow$ | **Input Length** $\downarrow$ | **Output Length** $\downarrow$ |
|---|---|---|---|
| Direct Few-Shot | 56.9 ($\pm$ 1.2) | 531.7 ($\pm$ 10.1) | 0.0 ($\pm$ 0.0) |
| CoT Few-Shot | **62.7** ($\pm$ 1.3) | 1190.5 ($\pm$ 13.2) | 143.0 ($\pm$ 2.6) |
| STaR | **62.3** ($\pm$ 1.2) | **76.6** ($\pm$ 1.3) | 110.8 ($\pm$ 1.7) |
| Metareasoning | **62.4** ($\pm$ 1.3) | **76.6** ($\pm$ 1.3) | **86.2** ($\pm$ 2.0) |

Table 1: Comparison of different methods based on accuracy and length metrics, averaged across datasets, (means with 95% confidence intervals). The overall performance remains comparable, despite the decreased number of generated tokens.

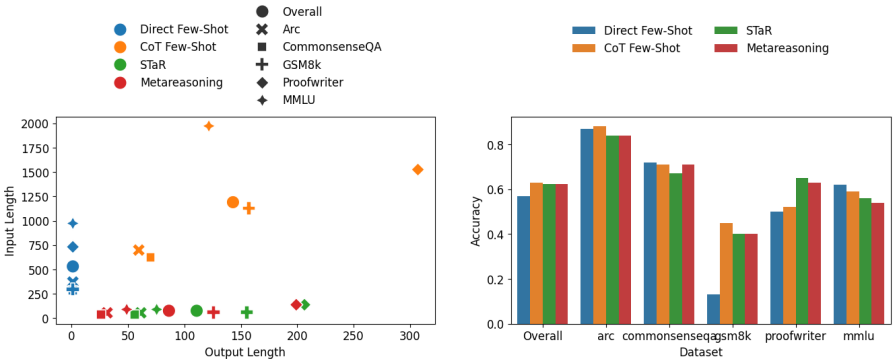


Figure 1: **Length and accuracy** On the left, the distribution of input and output lengths across various modalities is shown, organized by dataset. Our method (Metareasoning) eliminates the need for few-shot prompting, resulting in fewer input tokens and a lower output token count.

## 5 Conclusion

We have used rational metareasoning — the adaptive use of cognitive resources — to optimize reasoning in large language models (LLMs). Empirically, this approach reduces computational costs while maintaining comparable performance. However, we note some limitations and future directions. First, we focus on efficiency rather than task performance, and it remains to be seen whether our approach can be extended to improve task performance. Notably, our work demonstrates how cognitively inspired reward functions can equip LLMs with desirable inference-time properties. The flexibility of this method suggests it could be integrated with instruction tuning to further enhance efficiency. Additionally, by customizing the utility measure in the reward function, this strategy can guide models toward achieving specific, measurable improvements while still conserving computational resources.

## References


[1] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, et al. Gpt-4 technical report, 2024.

[2] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways, 2022.

[3] Alex de Vries. The growing energy footprint of artificial intelligence. *Joule*, 7(10):2191–2194, 2023.

[4] Roberto Verdecchia, June Sallou, and Luís Cruz. A systematic review of green <scp>ai</scp>. *WIREs Data Mining and Knowledge Discovery*, 13(4), June 2023.

[5] Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. Efficient large language models: A survey, 2024.

[6] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[7] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023.

[8] Walter Krämer. Kahneman, d. (2011): Thinking, fast and slow. *Stat. Pap. (Berl)*, 55(3):915–915, August 2014.

[9] Stuart J. Russell. Rationality and intelligence. *Artificial Intelligence*, 94(1):57–77, 1997. Economic Principles of Multi-Agent Systems.

[10] Falk Lieder and Thomas L. Griffiths. Strategy selection as rational metareasoning. *Psychological Review*, 124(6):762–794, November 2017.

[11] Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, May 1991.

[12] Falk Lieder, Amitai Shenhav, Sebastian Musslick, and Thomas L. Griffiths. Rational metareasoning and the plasticity of cognitive control. *PLOS Computational Biology*, 14(4):e1006043, April 2018.

[13] Thomas L Griffiths, Frederick Callaway, Michael B Chang, Erin Grant, Paul M Krueger, and Falk Lieder. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Current Opinion in Behavioral Sciences*, 29:24–30, October 2019.

[14] Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search, 2017.

[15] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022.

[16] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

[17] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge, 2019.

[18] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[19] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language, 2021.

[20] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021.

[21] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

[22] Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H. Chi, and Denny Zhou. Large language models as analogical reasoners, 2024.

[23] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023.

[24] Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V Le, and Denny Zhou. Take a step back: Evoking reasoning via abstraction in large language models, 2024.

[25] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve, 2022.

[26] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023.

[27] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2024.

[28] Alex Graves. Adaptive computation time for recurrent neural networks, 2017.

[29] Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder, 2021.

[30] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers, 2019.

[31] Amirkeivan Mohtashami, Matteo Pagliardini, and Martin Jaggi. Cotformer: More tokens with attention make up for less depth, 2023.

[32] Thomas L Griffiths, Frederick Callaway, Michael B Chang, Erin Grant, Paul M Krueger, and Falk Lieder. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Curr. Opin. Behav. Sci.*, 29:24–30, October 2019.

[33] Thomas L. Griffiths. Understanding human intelligence through human limitations. *Trends in Cognitive Sciences*, 24(11):873–883, November 2020.

[34] Alex Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. Teaching large language models to reason with reinforcement learning, 2024.

[35] Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling, 2023.

[36] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models, 2024.