ON REWARD FUNCTIONS FOR SELF-IMPROVING GENERAL-PURPOSE REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Prompting a Large Language Model (LLM) to output Chain-of-Thought (CoT) reasoning improves performance on complex problem-solving tasks. Further, several popular approaches exist to "self-improve" the abilities of LLMs to use CoT on tasks where supervised (question, answer) datasets are available. However, an emerging line of work explores whether self-improvement is possible without supervised datasets, instead utilizing the same large, general-purpose text corpora as used during pre-training. These pre-training datasets encompass large parts of human knowledge and dwarf all finetuning datasets in size. Self-improving CoT abilities on such general datasets could enhance reasoning for any general-purpose text generation task, and doing so at pre-training scale may unlock unprecedented reasoning abilities. In this paper, we outline the path towards self-improving CoT reasoning at pre-training scale and address fundamental challenges in this direction. We start by framing this as a reinforcement learning problem: given the first n tokens from a large pre-training corpus, the model generates a CoT and receives a reward based on how well the CoT helps predict the following m tokens. We then investigate a fundamental question: What constitutes a suitable reward function for learning to reason during general language modelling? We outline the desirable qualities of such a reward function and empirically demonstrate how different functions affect what reasoning is learnt and where reasoning is rewarded. Using these insights, we introduce a novel reward function called Reasoning Advantage (RA) that facilitates self-improving CoT reasoning on free-form question-answering (QA) data, where answers are unstructured and difficult to verify. Equipped with a suitable reward function, we explore the optimisation of it on general-purpose text using offline RL. Our analysis indicates that future work should investigate more powerful optimisation algorithms, potentially moving towards more online algorithms that better explore the space of CoT generations.

1 INTRODUCTION

Large Language Models (LLMs) have become increasingly effective at solving complex reasoning tasks (Huang & Chang, 2022; Kojima et al., 2023; Wei et al., 2023; Lee et al., 2024; Zhang et al., 2024b; Pang et al., 2024; Havrilla et al., 2024b). A key driver of this success has been Chain-of-Thought (CoT) reasoning (Wei et al., 2023), whereby a model is prompted, and perhaps explicitly trained, to output a step-by-step "thought process" before arriving at a final answer. Thus, simply appending *"Let's think step-by-step"* (Kojima et al., 2023) to the input of an LLM when asking questions has become a mainstay of language model prompting techniques.

Whilst some CoT reasoning ability emerges naturally from pre-training on unstructured web-text data (Fu et al., 2023), it is through further finetuning on curated datasets (Saparov & He, 2023) and Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) that CoT becomes such a powerful tool. Considerable effort is currently being invested in curating large-scale (question, CoT, answer) datasets for a wide range of tasks (Cobbe et al., 2021a; Saparov & He, 2023; Liu et al., 2023). Moreover, models are increasingly being employed "in the loop" to generate initial reasoning traces or refine existing ones (Zhang et al., 2024a), and in certain domains like mathematics, it is possible to further automate dataset generation by sampling many CoTs and selecting those which lead to ground-truth answers (Zelikman et al., 2022). However, despite these recent advancements, there are significant limitations to relying on curated datasets for improving CoT

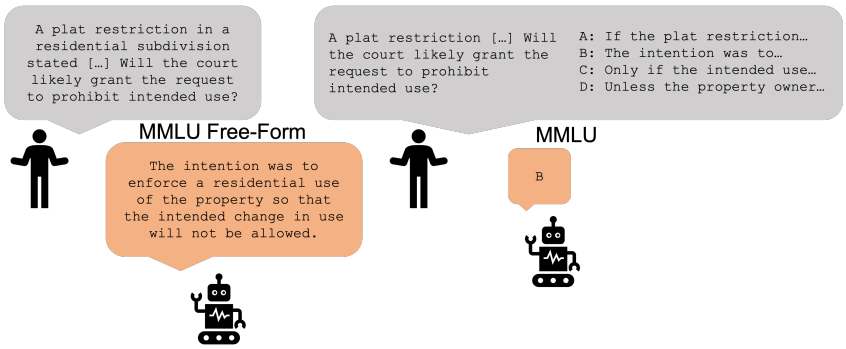


Figure 1: Example question and correct answer in MMLU-FREE-FORM, a novel dataset we create to study general-purpose reasoning. The question, answer pair is produced by removed the multiple-choice answers from the popular MMLU dataset (Hendrycks et al., 2020). The free-form text answer could be phrased many alternative ways, which invalidates the use of accuracy without an external verifier. MMLU-FREE-FORM opens up the ability for efficient research into self-improving general-purpose reasoning with academic-scale compute.

abilities. It is becoming increasing difficult to curate sufficiently challenging, large-scale (question, CoT, answer) datasets across the wide array of domains that today's general models can tackle. For example, a popular benchmark of graduate-level questions in biology, physics, and chemistry costs over $120,000 to produce and required thousands of expert hours (Rein et al., 2023).

To address these limitations, an emerging line of work explores self-improving CoT generation in a self-supervised setting, leveraging the large, unstructured text corpora used for pre-training (Zelikman et al., 2024). In this setting, the LLM learns to produce CoT reasoning for the task of next-token prediction: given n tokens from the pre-training corpus, the model generates a CoT and receives a reward based on how well the CoT helps predict the following m tokens. This is an exciting prospect, as we have trillions of tokens of unstructured text that encompass much of human knowledge. Learning to self-improve CoT reasoning on pre-training scale data might overcome the data availability bottleneck present in current curation-based self-improvement methods, and opens the door towards *compute-only scaling* of language model capabilities.

While there have been some initial efforts towards self-improving CoT during pre-training, we investigate a fundamental problem in this emerging line of work: What constitutes a suitable reward function for learning to reason during general language modelling? In Section 4, we outline the desirable qualities of such a reward function, and in Section 5.1, we empirically investigate how different functions affect:

1. *What* reasoning is rewarded—the ability to distinguish effective CoT reasoning.
2. *Where* reasoning is rewarded—the ability to pick out useful locations for generating CoT reasoning.

Our investigations reveal critical shortcomings in many commonly used reward functions, including an inability to differentiate between meaningful CoT reasoning and random word sequences (poor *what*: failing to reward effective reasoning), as well as a tendency to incentivize reasoning at locations where predicting following tokens is trivial (poor *where*: inability to pick out useful locations for reasoning). Drawing on these insights, we introduce a novel reward function called Reasoning Advantage (RA), an augmentation of standard language modelling loss with improved ability to select *what* and *where* to contemplate. We demonstrate that RA addresses many limitations observed in other reward functions and satisfies the criteria we established for effective self-improvement of CoT abilities during general-purpose language modelling.

Next, we create a challenging "free-form" QA dataset called MMLU-FREE-FORM by adapting the popular MMLU dataset (Hendrycks et al., 2020) into an open-ended or "free-form" QA setting: removing its multiple-choice format and requiring models to generate full, unstructured answers, as shown in Figure 1. Self-improving CoT reasoning on MMLU-FREE-FORM is a step towards the ultimate goal of self-improving CoT reasoning on general-purpose text, since answers are now non-unique, in variety of formats and challenging to verify. Moreover, this dataset provides a higher

density of clear opportunities for CoT reasoning compared to typical pre-training corpora, as we know that reasoning is particularly beneficial when predicting answers to questions. For the purposes of our investigations, this enables more efficient study and comparison of reward functions. In Section 5.2, we demonstrate that optimizing for RA on MMLU-FREE-FORM using a simple offline RL algorithm leads to positive transfer on the GSM8K benchmark of grade-school math problems (Cobbe et al., 2021b) improving accuracy by nearly 7% when trained with RA, compared to barely 0.5% when trained with other reward functions.

Equipped with our novel Reasoning Advantage (RA) reward function, we conduct initial experiments on the ultimate goal of self-improving CoT abilities on general-purpose language modelling at pre-training scale. Specifically, we use the OpenWebMath dataset (Paster et al., 2023), a vast collection of 14.7 billion tokens of maths-heavy text. We report our findings in Appendix D, which indicate that the offline RL method employed on MMLU-FREE-FORM is not sufficient for optimizing reward in a more general, less structured pre-training setting. We present a discussion of the remaining challenges and outline critical research questions for future work to address on the path towards self-improving CoT reasoning at pre-training scale.

In summary, our main contributions are as follows:

- We establish desirable criteria of reward functions for self-improving CoT reasoning on general-purpose language modelling at pre-training scale.
- We provide empirical evidence demonstrating how different reward functions impact both the quality of CoT reasoning (*what* reasoning is rewarded) and the ability to pick out useful locations for generating CoT reasoning (*where* reasoning is rewarded).
- We introduce MMLU-FREE-FORM, an open-ended QA dataset that facilitates more efficient study of self-improving CoT reasoning.we demonstrate that RA is the only reward function which enables self-improvement of CoT reasoning on free-form QA data, a key step towards self-improving CoT at pre-training scale. Self-improving CoT reasoning on MMLU-FREE-FORM is a step towards the ultimate goal of self-improving CoT reasoning on general-purpose text.
- We propose Reasoning Advantage (RA), a novel reward function based on clipped normalized loss, and demonstrate that RA is the only reward function which enables self-improvement of CoT reasoning on MMLU-FREE-FORM, a key step towards self-improving CoT at pre-training scale.
- We explore the optimisation of RA on the pre-training, general-purpose OpenWebMath dataset (Paster et al., 2023) using offline RL and find that it isn't sufficiently powerful to escape the local maxima of extremely conservative CoT reasoning. This suggests future work should investigate more powerful optimisation algorithms. In particular, potentially moving towards more online algorithms that better explore the space of CoT generations. We open source all our code that runs on academic compute to facilitate future work on this.

2 BACKGROUND

CoT Reasoning Given n prefix tokens $\mathbf{p}$, performing CoT reasoning refers to an LLM $\mathcal{M}$ generating a sequence of reasoning tokens $\mathbf{r}$ before the m answer suffix tokens $\mathbf{s}$. The goal of generating CoT reasoning tokens before the final answer is to maximize $P_{\mathcal{M}}(\mathbf{s}|\mathbf{p}, \mathbf{r})$, the probably of the answer suffix tokens $\mathbf{s}$ conditioned on both the prefix $\mathbf{p}$ and the CoT reasoning tokens $\mathbf{r}$. The prefix-suffix pair can be any token sequence, ranging from question-answer pairs in mathematical datasets to arbitrarily split sentences from an unstructured text corpus.

Traditionally, CoT reasoning has been elicited by pretending few-shot examples of (question, CoT, answer) to the prefix. This approach relies the pattern-completion tendencies of LLMs to continue this structure for subsequent outputs. Alternatively, it has also become popular to elicit CoT reasoning by appending prompts like "Let's think step by step." to the prefix (e.g., to the end of input questions), especially for instruction-tuned models.

Self-improving CoT Reasoning as Reinforcement Learning Self-improvement refers to any process where an LLM is finetuned on self-generated data, leading to performance gains without

human intervention or assistance from larger models. This process can be framed as a Reinforcement Learning (RL) problem. In RL, an agent interacts with an environment by taking actions $a \in A$ in states $s \in S$ to maximize cumulative rewards. The agent receives a reward $R_t = R(s_t, a_t)$ after each action a_t and aims to learn a policy $\pi(a|s)$ that maximizes the expected cumulative discounted reward $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k}$, where $\gamma \in [0, 1]$ is the discount factor.

In the context of CoT generation, each token can be viewed as an action a_t, with the current string of generated tokens representing the state s_t so far. We focus on a sparse reward setting where rewards are 0 until CoT generation is complete, and with a discount factor $\gamma = 1$. The reward function maps the prefix $\mathbf{p}$, CoT reasoning tokens $\mathbf{r}$, and answer suffix $\mathbf{s}$ to a real number $R(\mathbf{p}, \mathbf{r}, \mathbf{s}) \in \mathbb{R}$, with higher rewards for CoTs that better predict the suffix. As long as this reward function doesn't require external intervention from humans or more powerful models, optimizing it through RL methods constitutes self-improving CoT reasoning.

Self-improving CoT Reasoning Using Supervised Datasets When a supervised dataset of (question, answer) pairs is available, accuracy can serve as a reward function:

$$R_{\text{acc}}(\mathbf{p}, \mathbf{r}, \mathbf{s}) = \begin{cases} 1 & \text{if } \arg\max_{\mathbf{s}'} P_{\mathcal{M}}(\mathbf{s}'|\mathbf{p}, \mathbf{r}) = \mathbf{s} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In this case, we can sample multiple CoTs and finetune on those that lead to correct answers (Dong et al., 2023; Zelikman et al., 2022). Iterating this process yields increasingly high-quality CoT generation, and this iterative self-improvement is equivalent to online reinforcement learning (Zelikman et al., 2022). There are also more complex methods, such as Process Reward Models (PRMs), which provide dense rewards for each step in a CoT and address credit assignment challenges (Ma et al., 2023; Wang et al., 2023; Havrilla et al., 2024b; Lightman et al., 2023).

Self-improving CoT Reasoning on General-Purpose, Unstructured Data This setting explores the possibility of self-improving CoT reasoning given only an unstructured corpus of text, without access to a curated dataset of (question, CoT, answer) or (question, answer) pairs. In this setting, the model generates and inserts intermediate CoT reasoning at various points in a sequence of tokens (for example, at various points in a web-document that shows how to apply the quadratic formula).

A key challenge in this setting is evaluating the performance of CoT reasoning tokens inserted into general-purpose text. The accuracy-based reward R_{acc} is ineffective here, as it would almost always be 0, providing minimal learning signal. Instead, language modelling performance—the log-likelihood of the suffix conditioned on the prefix and CoT—serves as a more natural starting point for a reward function:

$$R_{\text{loss}}(\mathbf{p}, \mathbf{r}, \mathbf{s}) = \log P_{\mathcal{M}}(\mathbf{s}|\mathbf{p}, \mathbf{r}) \tag{2}$$

The ultimate goal of our work is to advance the field towards this setting, enabling self-improving CoT reasoning on general-purpose, unstructured data. In this paper, we specifically focus on developing and analyzing reward functions to address the unique challenges posed by this unstructured environment.

3 Related Work

LLM Reasoning Various works have looked at improving the reasoning capabilities of LLMs. Rajani et al. (2019) improve the commonsense reasoning ability of language models by training on human explanations for commonsense problems. Nye et al. (2021) generate tokens in a "scratchpad" for intermediate computations when solving multi-step reasoning problems. On difficult algorithmic tasks, Pfau et al. (2024) show that LLMs can even be trained to leverage meaningless filler tokens under dense supervision, in place of legible CoTs. Further, theoretical analyses by Merrill & Sabharwal (2023) and Feng et al. (2024) show that CoT improves the expressivity of Transformers (Vaswani et al., 2017).

LLM Self-improvement Using Supervised Datasets Iterated learning approaches involve LLMs generating new outputs and using "successful" ones to improve generation quality (Anthony et al., 2017; Vani et al., 2021; Polu et al., 2022). Such methods have been applied to LLMs (Zelikman

et al., 2022; Huang et al., 2022; Chen et al., 2024). However, much of the research on LLM self-improvement has been limited to question-answer domains where accuracy is an appropriate success measure, such as multiple-choice questions or simple numeric answers. This limitation is evident in the policy gradient objective approximated by STaR (Zelikman et al., 2022): $\nabla J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot|x_i)}[1(\hat{y}_i = y_i) \cdot \nabla \log p_M(\hat{y}_i, \hat{r}_i|x_i)]$, which makes use of an indicator function with respect to ground truth labels. Clearly, this breaks down in settings where ground truth labels are not available, such as open-ended or "free-form" QA setting as well as general-purpose language modelling. Havrilla et al. (2024a) show that Expert Iteration (Anthony et al., 2017), a self-improvement method based on iterative Supervised Fine-Tuning (SFT), outperforms RL in their evaluations. Building on this, our work extends RAFT (Dong et al., 2023), which also uses iterative SFT, by introducing a new reward function called Reasoning Advantage (RA) for filtering synthetically generated CoTs.

Process Reward Models (PRMs) (Ma et al., 2023; Wang et al., 2023; Havrilla et al., 2024b; Lightman et al., 2023) have been used to enhance reasoning via Reinforcement Learning (RL) by rewarding individual problem-solving steps in a CoT. However, PRM training is computationally expensive, usually involving backtracking and resampling from specific points in the CoT, and these points from which to resample are usually determined by hard-coded heuristics such as new line breaks.

Self-supervised LLM Self-improvement Quiet-STaR (Zelikman et al., 2024) is concurrent work that looks to self-improve reasoning during general-purpose language modelling. Zelikman et al. generate a CoT at every location of a general-purpose text document, using the negative cross-entropy loss on the suffix tokens as a reward. They employ REINFORCE (Williams, 1992) to optimise the loss of the suffix $\mathbf{s}$ given a prefix $\mathbf{p}$ and a reasoning trace $\mathbf{r}$, with a baseline for variance reduction. Importantly, performing CoT reasoning at every token is very computationally expensive, and therefore limits the length of CoT sequences that can be learnt. Thus, the reasoning learnt in Quiet-STaR is quite simple. Quiet-STaR provides important insights into *how* to optimise for reward on general-purpose text. Our work aims to take a step back and investigate *what* reward we should be optimising for in the first place.

4 Reward Functions for Self-improving General-Purpose Reasoning

In Section 2, we framed self-improving Chain-of-Thought (CoT) reasoning as a Reinforcement Learning (RL) problem. Given n tokens from a pre-training corpus (the prefix $\mathbf{p}$), the model generates a CoT $\mathbf{r}$ and receives a reward based on how well the CoT helps predict the following m tokens (the suffix $\mathbf{s}$). We previously introduced two choices of reward function: loss and accuracy. In this section, we explore other potential reward functions and their characteristics, from the perspective of facilitating self-improving CoT for general-purpose reasoning at pre-training scale.

An effective reward function should possess several key qualities. Primarily, it should reward high-quality reasoning over CoTs containing logical errors or just random characters. Moreover, for the purposes of *self-improving* CoT reasoning, the reward function must not depend on an external source of intelligence, such as using a more powerful LLM to verify the correctness of its CoT. In addition, for reasonable use at the pre-training scale, evaluating the function should be fast and ideally parallelizable (requiring a minimal number of model forward passes).

In this work, we do not consider using an LLM-as-judge to evaluate or "verify" CoTs. First, this setting is not suitable for self-improvement as it relies on an external source of intelligence. Further, while one could use the same model for generation and verification, this approach incurs too much computational overhead to apply to pre-training scale data. It requires the decoding of an answer to be verified against the ground truth and the verifier itself often needs to generate CoT tokens, both which make it too slow for application to pre-training scale data.

Thus, we focus on "loss-based" reward functions. These functions compute the token-by-token log-likelihood of the suffix tokens $\mathbf{s}_{0,\ldots,m-1}$, given the CoT $\mathbf{r}$ and prefix $\mathbf{p}$:

$$\begin{aligned}\log P(\mathbf{s}|\mathbf{p},\mathbf{r}) = &\log P(\mathbf{s}_0|\mathbf{p},\mathbf{r}) \\ &+ \log P(\mathbf{s}_1|\mathbf{p},\mathbf{r},\mathbf{s}_0) \\ &+ \log P(\mathbf{s}_2|\mathbf{p},\mathbf{r},\mathbf{s}_{0,1}) \\ &+ \ldots\end{aligned} \tag{3}$$

Therefore, the most basic reward function in this family is $R(\mathbf{p},\mathbf{r},\mathbf{s}) = \log P(\mathbf{s}|\mathbf{p},\mathbf{r})$. This family of reward functions offers several key advantages. First, they are computationally efficient, as they can be evaluated by an autoregressive model in a single forward pass and can be parallelized across a batch of CoTs. Second, they do not require access to any external form of intelligence. Last, and most crucially, this family of functions does not rely on an exact match to the answer suffix, allowing for multiple valid answers and accommodating ambiguity in formatting.

Now, while there are many possible ways to augment the basic loss-based reward function $R(\mathbf{p},\mathbf{r},\mathbf{s}) = \log P(\mathbf{s}|\mathbf{p},\mathbf{r})$, we focus our analysis on two key modifications below: clipping and the incorporation of a baseline.

Clipping We apply a clipping operation to the token-level log probabilities $\log P(\mathbf{s}_i|\mathbf{p},\mathbf{r},\mathbf{s}_{0,\ldots,i-1})$, clamping the minimum value to $-\epsilon$. This constrains the loss contribution of each suffix token to the range $[-\epsilon, 0)$. In Section 5.1, we demonstrate that this clipping mechanism helps reward functions distinguish between well-formed CoTs containing a few logical errors and degenerate CoTs resembling random tokens.

Baseline Incorporation We investigate three types of baselines:

1. Average reward: $\frac{1}{n}\sum_{i=1}^{n} R(\mathbf{p},\mathbf{r_i},\mathbf{s})$, where $\mathbf{r_i}$ are multiple generated CoTs.
2. Empty CoT reward: $R(\mathbf{p}, \text{“ ”}, \mathbf{s})$, where the CoT is an empty string.
3. Random CoT reward: $R(\mathbf{p}, \mathbf{r_{random}}, \mathbf{s})$, where $\mathbf{r_{random}}$ is a sequence of random tokens.

We explore incorporating these baselines both with normalization $(R-B)/B$ and without normalization $R-B$, where R is the reward and B is the baseline value. A full list and derivation of the reward functions we investigate can be found in Appendix A.

Delta Loss We define Delta Loss as the difference between the loss in Equation 3 and the "Empty CoT reward" baseline. That is, $R_{\text{Delta Loss}} = R(\mathbf{p},\mathbf{r},\mathbf{s}) - R(\mathbf{p}, \text{“ ”}, \mathbf{s})$.

Reasoning Advantage (RA) Among the various combinations of clipping and baseline incorporation, we identify a particularly effective reward function which we term Reasoning Advantage (RA). RA uses clipping and the normalized "Empty CoT reward" baseline. It satisfies all of the identified criteria of reward functions for self-improving CoT reasoning at pre-training scale, which are summarized in Table 1. In Section 5.1, we empirically demonstrate that RA can best distinguish effective CoT and best pick out useful locations for CoT reasoning. In Section 5.2, we demonstrate that RA is the only reward function which enables self-improvement of CoT reasoning on free-form QA data, a key step towards self-improving CoT at pre-training scale.

5 Experiments

5.1 Reward Functions for Selecting What & Where to Reason

In this section, we empirically investigate a fundamental problem when self-improving CoT reasoning during pre-training: **What constitutes a suitable reward function for learning to reason during general language modelling?** Building on the desirable qualities of reward functions outlined in Section 4, we empirically investigate how different reward functions affect *what* and *where* reasoning is rewarded. Our two experiments reveal critical shortcomings in commonly used reward functions and demonstrate the advantages of our novel Reasoning Advantage (RA) function in addressing these limitations.

Criteria	Accuracy	Loss	Loss with baseline	RA	LLM-as-judge
Uses no external intelligence	Yes	Yes	Yes	Yes	Yes[2]
Rewards good reasoning over random	Yes	**No**	**No**	Yes	Yes
Robust to multiple choices in answer	**No**	Yes	Yes	Yes	Yes[3]
Robust to answer perplexity	Yes	**No**	**No**	Yes	Yes
Fast and parrallelisable	No[1]	Yes	Yes	Yes	**No**

Table 1: To what extent different reward functions meet our criteria. By RA, we mean loss augmented with clipping and the no CoT baseline, as defined in Appendix A. [1]whilst we do derive a generation free variant 'expected accuracy' in appendix A that is as fast as loss based methods, the variant of accuracy used widely through the literature requires answers to be sampled, and so is slow. [2]Whilst acting as a verifier may be possible for larger models under heavy prompting, we found it difficult to consistently verify solutions with the 7B models we used for generation and finetuning. [3]Again, whilst this may be possible with more work, we found it very difficult to have models consistently grade CoTs that yielded answers close to, but not exactly, the right answer.

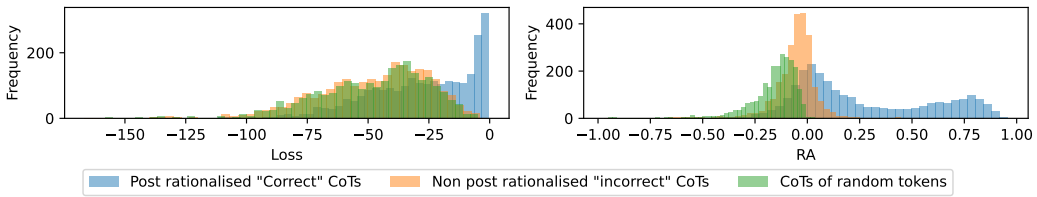
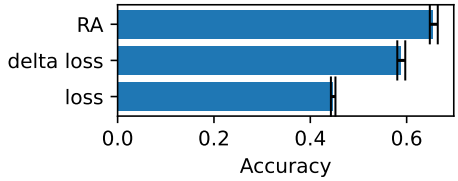


Figure 2: *What to contemplate*: Loss and RA differ in how they score different classes of CoT. Note how the clipping used in RA has removed the overlap between "incorrect" and "random" tokens. Scores for all CoTs are all nicely normalised into [-1, 1] which we posit is better for learning.

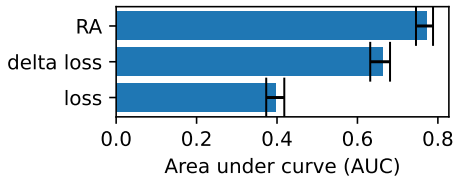
***What* reasoning is rewarded** This first experiment evaluates the ability of different reward functions to distinguish between three categories of CoTs: correct, incorrect, and randomly generated. We select 1,000 prefix-suffix pairs from random locations in the FineWeb text corpus (Penedo et al., 2024). Then, for each pair, we generate the three types of CoT. "Correct" CoTs are generated using GPT-4o with "post-rationalization." That is, by showing GPT-4o both the prefix and suffix, but instructing the model to generate a CoT without revealing its knowledge of the suffix, similar to the technique used in STaR (Zelikman et al., 2022). "Incorrect" CoTs are generated by GPT-4o without post-rationalization. While these CoTs often exhibit sophisticated reasoning, they typically don't predict the exact suffix as well as the "correct" CoTs, hence their classification as "incorrect" for the purposes of this experiment. Finally, random CoTs consist of strings of random words and serve as our baseline. The goal is to evaluate how well different reward functions can rank these CoT types, with the ideal ordering: correct $>$ incorrect $>$ random.

For each reward function, we sort the CoTs by their computed reward and classify the top third as "correct," the middle third as "incorrect," and the bottom third as "random." Figure 3a shows the classification accuracy for RA, delta loss, and loss. In our experiments, RA performs best out of the three. Table 2 in the appendix presents the results for the full list of evaluated reward functions, with the row for RA in bold. Notice that while the no normalization version of RA performs just slightly better, the normalized version significantly outperforms all other functions in the *where* experiments below, which is why we pick the normalized version.

In contrast, selecting by loss on the suffix alone performs poorly. The histogram in Figure 2 illuminates that this poor performance is primarily due to an inability to distinguish between "incorrect" and "random" CoTs. This is the primary benefit of clipping - when we simplify the task to a binary classification of "correct" vs. non-correct CoTs we found that non-clipping methods perform comparably well. The full results in Table 2 in Appendix B shows that the average reward baseline as used by Quiet-STaR performs poorly in this setting. This is due to a lack of variation in reward over different CoTs.



(a) *What to contemplate*: The accuracy of each reward function to distinguish between "correct", "bad" and "random" reasoning



(b) *Where to contemplate*: The AUC of each reward function at classifying between "good" and "bad" locations to reason in the text.

Figure 3: The ability of different reward functions to choose when and where to produce general-purpose CoTs. For a full table of results see Appendix B. RA outperforms all commonly used reward functions.

***Where* reasoning is rewarded** Next, we explore how the choice of reward function affects where in a document it is most beneficial to produce CoT reasoning. We select 1,000 problems from each of GSM8K (Cobbe et al., 2021b), CSQA and MMLU (Hendrycks et al., 2020) and format them as a single text string. We then produce 4 prefix, suffix pairs by splitting the at 4 different locations in each problem: 1) halfway through the question 2) after the question but before the multiple choice answers have been given. Many questions are still quite ambiguous at this stage 3) After the multiple choice answers. We argue that this location is the most optimal place to contemplate. 4) Halfway through the answer. Given the multiple choice answers have already given all the possible suffixes, it is trivial to predict the suffix at this location.

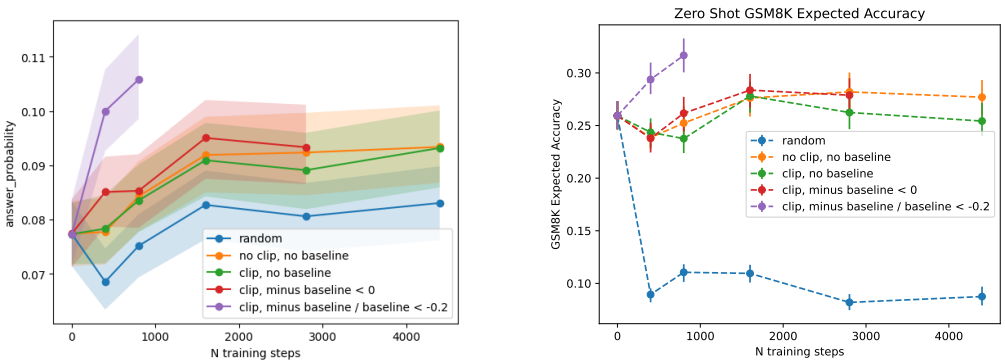
The experiment aims to replicate the finding that in general-purpose, unstructured text some suffixes are far harder to predict than others (it is almost impossible to predict the suffix halfway through the question), but only some of these locations are worth spending time reasoning about.

As with the *what to contemplate* experiment, we sort all CoTs for all locations by reward score. Since there is no obvious ranking between locations, we look at the binary classification problem of selecting the "after multiple choice answers" location, that we take to be optimal. Table 3 in Appendix B gives the area under curve (AUC) metric for various reward functions, and is summarised by Figure 3b. We see that the methods with a "not contemplating" baseline perform best, with the clipping variants performing marginally better. Unlike the *what to contemplate* experiment, the random baseline performs very poorly. This is likely due to the model used to compute the rewards performing unexpectedly when presented with a random contemplation. Again loss performed poorly, due the fact that, regardless of the CoT, the halfway through the answer is easy to predict, and thus these CoTs all scored highly. See Figure 7 in Appendix B to see this.

Whilst no single metric was stand-out across both experiments, we can conclude that using a baseline is beneficial. The most promising baseline is the "not contemplating" baseline, since it requires only 1 CoT per prefix suffix pair (compared to multiple for an average across CoTs) and performed better than the random contemplation baseline in the *where to contemplate* experiment. Clipping was often beneficial, almost never harmful and requires minimal additional computation. It is yet to be seen whether distinguishing between random CoTs and incorrect CoTs is actually beneficial when optimising.

5.2 Learning to Reason on Free-Form QA Data

Equipped with the analysis of different reward functions above, we now look to optimise them on reasoning tasks. We produce a new "free-form" question-answering (QA) dataset called MMLU-Free-Form, that takes the popular MMLU (Hendrycks et al., 2021) dataset and augments it to be more similar to the full general-purpose text setting. We do this instead of diving straight into the full "general-purpose reasoning" problem primarily to allow us to compare reward functions at a smaller scale. We know that reasoning abilities on MMLU can be improved with a few 1,000 CoTs in the original setting, compared to an unknown and likely much larger scale required for general-purpose contemplation. Furthermore, the density of locations for which it is desirable to reason in MMLU is far higher than in general-purpose text, such as OpenWebMath (Paster et al., 2023). This allows us to compare and ablate different loss functions in a far more compute and time efficient manner.



(a) MMLU Test Performance

(b) GSM8K Transfer Performance

Figure 4: Learning to reason on Free-Form QA Data. The methods presented each train for a different number of steps because they yield different amounts of data after filtering. All baselines here are the "Empty CoT" reward.

We produce MMLU-FREE-FORM by simply taking the original MMLU questions, but not giving the model access to the original set of multiple choice answers. This induces many of the challenges found in general-purpose contemplation. Firstly, some problems become almost impossibly difficult to answer ("Which of the following is the correct method of multiple 32 x 18?"), as is the case with many next-token prediction problems in general-purpose text. Secondly, answers become vastly different in difficulty to exactly predict, for one because they are vastly different lengths. Finally, there are often many ways to say the same answer ("Henry VIII had 6 wives" vs "In total there were 6 different woman who were married to Henry the Eighth"), and without the answer list it is not obvious which should be used.

Given a reward function, we optimise for it using a simple offline method. We generate a large number of CoTs, finetune on the highest scoring ones. This allows us to directly and efficiently compare reward functions, since the initial generated CoTs are the same for each reward function.

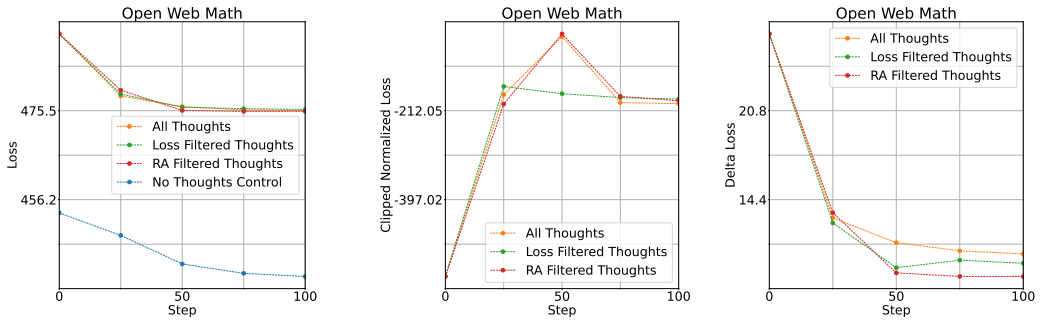
We evaluate the performance of our optimised models on the heldout test set of MMLU-Free-Form. We look at the probability of the answer given the question and the CoT. This is also known as "expect accuracy", since it is the expected number of times you would exactly generate the ground truth correct answer if you sampled a larger number of answers conditioned on the question and thought. We produce 95% confidence intervals through bootstrapping (LaFlair et al., 2015).

We see that RA alone is able to increase the answer probability on the test set. Selecting CoTs by loss, delta loss or simple random selection only improves test performance by a few percent, and starts to plateau quickly with more steps. We were only able to train for 1,000 steps with RA, since only 1,000 steps worth of generated CoTs were above the threshold of 0.2.

The reasoning learnt by training on the MMLU-FREE-FORM CoTs transfers to GSM8K math problems. We see a performance boost of nearly 7% with RA, compared to barely 0.5% for other reward functions at the same amount of training steps. A full breakdown of in-domain MMLU performance is shown in Figure 7 in Appendix C.

6 CHALLENGES AND FUTURE DIRECTIONS

To chart a course for future work, we present an exploratory experiment that extends the self-improving CoT methodology from Section 5.2 to a general-purpose language modeling dataset. Specifically, we investigate optimizing for Reasoning Advantage (RA) on the pre-training corpus OpenWebMath (Paster et al., 2023). We employ the same offline RL procedure we used in our experiments on MMLU-FREE-FORM: generating a large batch of CoTs and finetuning the model on those with the highest reward scores. This is an attractive place to start, as unlike for online training of LLMs, high performance open-source libraries exist for both language model generation and Su-



(a) LM Loss with thoughts (b) RA (c) Delta Loss

Figure 5: The language modelling performance related metrics for an experiment that attempts to optimise language loss with CoTs on Open-Web-Math.

pervised FineTuning (SFT), making offline RL an approachable and scalable method for researchers without industry-scale compute.

However, as illustrated in Figure 5, we find that the offline RL training algorithm which succeeded on MMLU-FREE-FORM is not sufficiently powerful to escape the local optimum of extremely conservative CoT reasoning on general-purpose, pre-training text. Examples of these CoTs are shown in Appendix E. In fact, the offline RL algorithm we use is so inadequate at optimisation in this regime that optimising for RA actually results in models that generate CoTs with lower RA values than models optimised with loss (Figure 5. The failure to escape this local maxima is likely due to a lack of diversity in the offline training corpus - the CoTs in this initial dataset are generated using a single prompt. While we are confident in RA's ability to identify good reasoning when it exists (given the results in Section 5.1), it is still difficult to a-priori prompt the model to create high performing CoTs. Only 0.01% of the generated CoTs for OpenWebMath achieve a reward above 0.2 (which from our experience is a decent heuristic for a "good reasoning" threshold). This suggests that the quality of the generated CoTs are simply too low to be useful on OpenWebMath.

There are numerous ways to increase the diversity of CoTs explored and learnt. Using Quality-Diversity (Mouret & Clune, 2015) or other evolutionary techniques (Fernando et al., 2023; Samvelyan et al., 2024) could lead the generation of a more diverse dataset. Whilst exploration may alternatively be facilitated by instead using online RL, the only existing method for doing so requires 8×H100s to train (Zelikman et al., 2024). We believe that the computational feasibility of generating CoTs in large, offline batches and performing supervised fine-tuning is advantageous under more limited compute constraints. To facilitate future research we therefore open-source the code for our offline approach, which can be run on an academic compute budget and enables isolating the individual components of this problem (i.e., generating good CoTs and learning from good CoTs). Furthermore we release the dataset MMLU-FREE-FORM, which serves as a middle ground between the settings of general purpose language modelling and Q&A. We believe it provides a suitable testing ground for further research on CoT-assisted language modelling.

7 CONCLUSION

In this paper, we outline a path towards self-improving CoT reasoning at pre-training scale and address fundamental challenges in this direction. We frame this as a reinforcement learning problem and investigate a fundamental question: What constitutes a suitable reward function for learning to reason during general language modelling? We outline the desirable qualities of such a reward function and empirically demonstrate how different functions affect what reasoning is learnt and where reasoning is rewarded. Using these insights, we introduce a novel reward function called Reasoning Advantage (RA) which is the only reward function that facilitates self-improving CoT reasoning on free-form question-answering (QA) data, where answers are unstructured and difficult to verify. Finally, we present an exploratory experiment optimizing RA on general-purpose, pre-training text and discuss the obstacles towards self-improving CoT reasoning at pre-training scale, as well as directions for future work. We open source all our code that runs on academic compute to facilitate future work.

REFERENCES

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. *Advances in neural information processing systems*, 30, 2017.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

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, 2021a. URL https://arxiv.org/abs/2110.14168.

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

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment, 2023.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36, 2024.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution, 2023. URL https://arxiv.org/abs/2309.16797.

Yao Fu, Litu Ou, Mingyu Chen, Yuhao Wan, Hao Peng, and Tushar Khot. Chain-of-thought hub: A continuous effort to measure large language models' reasoning performance, 2023. URL https://arxiv.org/abs/2305.17306.

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. *arXiv preprint arXiv:2403.04642*, 2024a.

Alex Havrilla, Sharath Raparthy, Christoforus Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, and Roberta Railneau. Glore: When, where, and how to improve llm reasoning via global and local refinements. *arXiv preprint arXiv:2402.10963*, 2024b.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

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

Geoffrey T LaFlair, Jesse Egbert, and Luke Plonsky. A practical guide to bootstrapping descriptive statistics, correlations, t tests, and anovas. In *Advancing quantitative methods in second language research*, pp. 46–77. Routledge, 2015.

Seungpil Lee, Woochang Sim, Donghyeon Shin, Sanha Hwang, Wongyu Seo, Jiwon Park, Seokki Lee, Sejin Kim, and Sundong Kim. Reasoning abilities of large language models: In-depth analysis on the abstraction and reasoning corpus. *arXiv preprint arXiv:2403.11793*, 2024.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. Logicot: Logical chain-of-thought instruction-tuning, 2023. URL https://arxiv.org/abs/2305.12147.

Qianli Ma, Haotian Zhou, Tingkai Liu, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. Let's reward step by step: Step-level reward model as the navigators for reasoning. *arXiv preprint arXiv:2310.10080*, 2023.

William Merrill and Ashish Sabharwal. The expresssive power of transformers with chain of thought. *arXiv preprint arXiv:2310.07923*, 2023.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites, 2015. URL https://arxiv.org/abs/1504.04909.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models, 2021.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*, 2024.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. *arXiv preprint arXiv:2310.06786*, 2023.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Jacob Pfau, William Merrill, and Samuel R Bowman. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*, 2024.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*, 2022.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning, 2019.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Mikayel Samvelyan, Sharath Chandra Raparthy, Andrei Lupu, Eric Hambro, Aram H. Markosyan, Manish Bhatt, Yuning Mao, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, Tim Rocktäschel, and Roberta Raileanu. Rainbow teaming: Open-ended generation of diverse adversarial prompts, 2024. URL https://arxiv.org/abs/2402.16822.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought, 2023. URL https://arxiv.org/abs/2210.01240.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. URL https://arxiv.org/abs/2302.04761.

Ankit Vani, Max Schwarzer, Yuchen Lu, Eeshan Dhekane, and Aaron Courville. Iterated learning for emergent systematicity in vqa. *arXiv preprint arXiv:2105.01119*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Y Wu, and Zhifang Sui. Math-shepherd: A label-free step-by-step verifier for llms in mathematical reasoning. *arXiv preprint arXiv:2312.08935*, 2023.

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.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D Goodman. Star: Self-taught reasoner bootstrapping reasoning with reasoning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, pp. 15476–15488, 2022.

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*, 2024.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b, 2024a. URL https://arxiv.org/abs/2406.07394.

Yadong Zhang, Shaoguang Mao, Tao Ge, Xun Wang, Adrian de Wynter, Yan Xia, Wenshan Wu, Ting Song, Man Lan, and Furu Wei. Llm as a mastermind: A survey of strategic reasoning with large language models. *arXiv preprint arXiv:2404.01230*, 2024b.

A FORMAL DEFINITIONS

We look at the following metrics for evaluation of intermediate contemplation for next token prediction, that is, given a prefix set of tokens $p = p_1, ..., p_n$, a generated set of intermediate reasoning tokens r and a suffix of m tokens to predict $s = s_1, ..., s_m$, produce a score $R(p, r, s) \in \mathbb{R}$. We use $P(s_0|p + r)$ to denote the probability distribution over all tokens on the first token of the suffix.

1. **Accuracy (using generation)**: Generate, such as through sampling or via greedy decoding, k continuations $\hat{s_1}, ..., \hat{s_k}$ of length n_s from the input $p + r$.

$$R_{\text{accuracy using generation}} = \frac{1}{k} \sum_{i=1}^{k} \mathbb{I}[\hat{s_i} = s] \tag{4}$$

2. **Accuracy (generation free)**: Accuracy using generation requires at least n_s forward passes. Instead, one can leverage the autoregressive nature of transformers to obtain the probability distribution over next tokens for the entire answer simultaneously. That is input the model $p + r + s$ and obtain $P(\hat{s_0}|p + r), P(\hat{s_1}|p + r + s_0), ...$ with one forward pass. Looking at whether the argmax of this distribution is s is equivalent to accuracy above using greedy decoding, and taking $P(s|p + r)$

$$R_{\text{expected greedy accuracy}} = \Pi_{i=1}^{n_s} \mathbb{I}[\arg\max(P(\hat{s_i}|p + r + s_{:i})) = s_i] \tag{5}$$

$$R_{\text{expected accuracy}} = \Pi_{i=1}^{n_s} P(\hat{s_i}|p + r + s_{:i}) \tag{6}$$

3. **Loss**: We use the cross entropy loss over tokens, i.e:

$$R_{\text{cross entropy loss}} = -\sum_{i=1}^{n_s} \log(P(\hat{s_i}|p + r + s_{:i})) \tag{7}$$

4. **Delta Loss:** The difference in cross entropy between using and not using the reasoning.

$$R_{\text{delta cross entropy loss}} = -\sum_{i=1}^{n_s} \log(P(\hat{s_i}|p + r + s_{:i})) - -\sum_{i=1}^{n_s} \log(P(\hat{s_i}|p + s_{:i})) \tag{8}$$

5. **Normalised Delta loss:** Different answers have varying levels of inherent predictability. Thus desirable values for loss or delta loss can vary massively. To account for this, we divide by the answer likelihood without reasoning.

$$R_{\text{normalised delta cross entropy loss}} = R_{\text{delta cross entropy loss}} / -\sum_{i=1}^{n_s} \log(P(\hat{s_i}|p + s_{:i})) \tag{9}$$

6. **Clipped variants**: We evaluate loss, delta loss and normalised delta loss with clipping applied to the token log probabilities to prevent large values dominating. Our final results leverage $\epsilon = -3$. For example

$$R_{\text{clipped loss}} = -\sum_{i=1}^{n_s} \max[\log(P(\hat{s_i}|p + r + s_{:i})), \epsilon] \tag{10}$$

7. Normalised clipped delta loss **(Reasoning Advantage)**: We combine the benefits of delta loss, normalisation and clipping into one metric.

8. **LLM-as-judge**: Generate, such as through sampling or via greedy decoding, k continuations $\hat{s_1}, ..., \hat{s_k}$ of length n_s from the input $p + r$. Let $M(p, r, \hat{s_i}, s_i)$ denote whether a model considers $\hat{s_i}$ to match be the correct answer of s_i.. Average over the k completions, i.e:

$$R_{\text{Model eval}} = \frac{1}{k} \sum_{i=1}^{k} M(p, r, \hat{s_i}, s_i) \tag{11}$$

For all metrics, we used a Mitral 7B (Jiang et al., 2023) model that has been finetuned on a small set of 1,000 GPT-4 generated COTs, filtered for correctness according to model eval, in order to start from a base model used to the format of

```
### Question: <question> ### Thought <reasoning> ### Answer: <response>.
```

Baseline	Clipping	Normalisation	Mean	q0.025	q0.975	Rank
none	none	none	44.6%	44.0%	45.4%	9
not contemplating	clipped	none	67.2%	65.7%	68.3%	3
not contemplating	**clipped**	**yes**	**66.3%**	**64.5%**	**67.8%**	**4**
not contemplating	none	none	58.3%	57.8%	58.9%	8
not contemplating	none	yes	58.8%	58.1%	59.8%	7
random contemplation	clipped	none	80.4%	79.7%	81.4%	1
random contemplation	clipped	yes	78.4%	77.8%	79.0%	2
random contemplation	none	none	60.9%	60.1%	62.7%	5
random contemplation	none	yes	60.9%	59.2%	63.1%	5
mean loss	clipped	none	30.8%	30.1%	31.7%	10
mean loss	clipped	yes	30.7%	29.9%	31.3%	11
mean loss	none	none	29.2%	28.7%	29.8%	13
mean loss	none	yes	30.7%	30.0%	31.7%	11

Table 2: What to contemplate. RA in bold.

Baseline	Clipping	Normalisation	Mean	q0.025	q0.975	Rank
none	none	none	39.4%	37.7%	40.8%	6
not contemplating	clipped	none	55.9%	52.5%	59.9%	4
not contemplating	**clipped**	**yes**	**77.0%**	**75.3%**	**79.0%**	**1**
not contemplating	none	none	64.4%	62.7%	67.0%	3
not contemplating	none	yes	73.0%	71.9%	74.3%	2
random contemplation	clipped	none	29.8%	28.2%	30.6%	9
random contemplation	clipped	yes	40.8%	38.9%	43.4%	5
random contemplation	none	none	27.9%	26.7%	28.8%	11
random contemplation	none	yes	27.3%	25.8%	28.6%	13
mean loss	clipped	none	27.7%	25.8%	29.2%	12
mean loss	clipped	yes	33.4%	32.5%	35.4%	7
mean loss	none	none	28.3%	26.5%	30.0%	10
mean loss	none	yes	32.1%	30.8%	33.4%	8

Table 3: Where to contemplate. RA in bold.

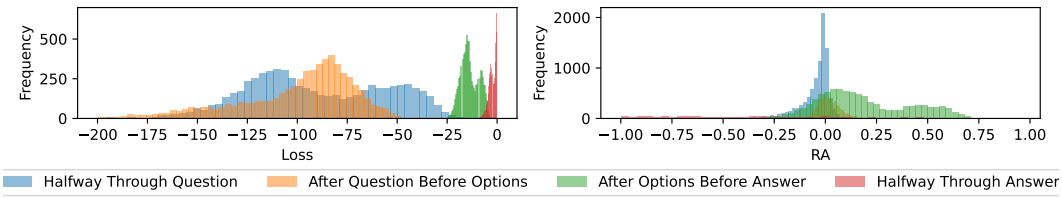


Figure 6: *Where to contemplate*: Loss and RA differ in how they score CoTs produced at different locations. Note how the incorporation of a baseline has successfully prevented the "halfway through answer" CoTs from being high scoring.

B Full results for where and what to contemplate

Tables 2 and 3 show a full table of results for the entire family of loss based reward functions.

C Breakdown of test-time reasoning on MMLU-Free-Form

A breakdown of test time performance by question style is shown in Figure 7.

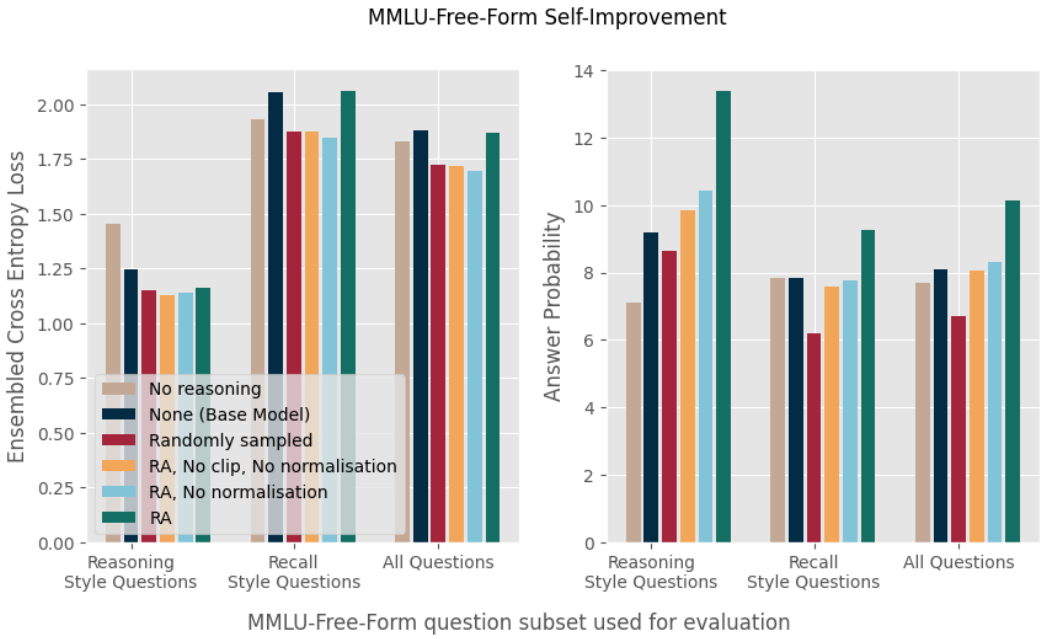


Figure 7: A breakdown of MMLU-FREE-FORM test time performance on a set of problems from the test set that we classify as "maths style reasoning" and "recall style reasoning". Ensembled cross-entropy loss is the average log-likelihood across multiple CoTs $\sum_i^N \log P(s|p, r_i)$

D OPTIMISING COT REWARD IN GENERAL LANGUAGE MODELLING

In this appendix we present initial explorations for optimising for a given reward metric. We optimise for it using a standard offline RL procedure known as RAFT (Dong et al., 2023), which is similar to that of (Schick et al., 2023). We randomly sample a 50,000 dataset of prefix, suffix pairs from the large unstructured open-web-math corpus [citation needed], and generate CoTs for each of them, using the Mistral-7b-Instruct model [citation needed]. We score by each reward function as described in [cite problem statement] and perform a standard supervised finetune of Mistral-7B on the top 3200 ranked CoTs (100 training steps at per device batch size 8 on 4 GPUs). The model used in the reward function is also Mistral-7B, finetuned on a small number of CoTs to introduce it to the "[Thought]...[/Thought]" syntax used to denote the start and end of each thought.

We included training on "all thoughts", that is a random selection of 3200 CoTs from the dataset as a baseline, as well as a second baseline of just training on just open-web-math without thoughts.

During training, we wish to evaluate both the utility of CoTs the model is able to generate and the models ability to select where to reason. To do this we take a holdout portion of open-web-math and scan through it, identifying locations in the text where the most likely next token according to the model is the start CoT "[Thought]" token. We then generate CoTs from these locations, insert them into text and compute a variety of reward functions to assess them.

Figure 5a plots the standard language modelling cross entropy loss on the holdout text using these evaluation thoughts. We exclude loss on inserted CoT tokens, but still use them to help model the document. We show that training on thoughts selected by Ra loss or even thoughts selected randomly decrease loss throughout training. None of our examined methods reduce loss faster than standard finetuning on open-web-math, which we show as a control. It is unsurprising that this baseline is hard to beat given how many evolutions our deep learning architectures have gone through to specialise at this task.

Figure 5c shows that during training on average thoughts are still hurting rather than helping the models ability to predict the suffix, since the change in loss compared to not contemplating is pos-

itive. The decrease in "lm loss with thoughts" is therefore likely due to improved abillity to model non-CoT tokens, rather than increased CoT ability.

E EXAMPLES OF "CONVERSATIVE" REASONING LEARNT DURING OPTIMISATION ON GENERAL PURPOSE TEXT

For GSM8K math problems:

```
Q: Richard lives in an apartment building with 15 floors. Each
    floor contains 8 units, and 3/4 of the building is occupied.
    What's the total number of unoccupied units In the building? [
    THOUGHT] The prefix provides information about an apartment
    building with 15 floors, each floor containing 8 units, and
    3/4 of the building occupied. The completion likely provides
    the total number of unoccupied units in the building, based on
     the given information. [/THOUGHT]
A: 30
```

For Open-Web-Math: Example 1:

```
In this article, by using norms($T$ and $C$), we present the
    concept of intuitionistic fuzzy implicative ideals,
    intuitionistic fuzzy closed implicative ideals and
    intuitionistic fuzzy commutative ideals of $BCI$-algebras.
    Some interesting results of them are given. Characterisations
    of implicative ideals, closed implicative ideals and
    commutative ideals of $BCI$-algebras by using them are
    explored. By using intersections, direct products and
    homomorphisms, some interesting results are obtained [THOUGHT]
     Consider the prefix, which introduces the concept of
    intuitionistic fuzzy implicative ideals in $BCI$-algebras. The
     completion likely provides a definition or description of
    these ideals, along with their properties and applications. [/
    THOUGHT] .
```

Example 2:

```
# Chapter 8 - Polynomials and Factoring - Chapter Review - 8-3 and
     8-4 Multiplying Binomials: 31

$9r^{2}-12r+4 [THOUGHT] The prefix provides a list of equations,
    each with a variable and a coefficient. The completion likely
    provides the solution to each equation, using the variable and
     coefficient to determine the value of the equation. The
    completion may also provide a step-by-step explanation [/
    THOUGHT] $

#### Work Step by Step

Simplify and write in standard form $(3r-2)^{2}$ Rewrite as: $(3r
    -2)(3r-2)$ Foil $9r^{2}-6r-6r+4$ Combine like terms $9r^{2}-12
    r+4$

After you claim an answer you'll have 24 hours to send in a draft.
     An editor will review the submission and either publish your
    submission or provide feedback.
```