REFINER: Reasoning Feedback on Intermediate Representations

Anonymous ACL submission

Abstract

Language models (LMs) have recently shown remarkable performance on reasoning tasks by explicitly generating intermediate inferences, e.g., chain-of-thought prompting. However, these intermediate inference steps may be inappropriate deductions from the initial context and lead to incorrect final predictions. Here we introduce REFINER, a framework for fine-tuning LMs to explicitly generate intermediate reasoning steps while interacting with a critic model that provides automated feedback on the reasoning. Specifically, the critic provides structured feedback that the reasoning LM uses to iteratively improve its intermediate arguments. Empirical evaluations of REFINER on three diverse reasoning tasks show significant improvements over baseline LMs of comparable scale. Furthermore, when using GPT-3.5 or ChatGPT as the reasoner, the trained critic significantly improves reasoning without fine-tuning the reasoner. Finally, our critic model is trained without expensive human-in-the-loop data but can be substituted with humans at inference time.

1 Introduction

Large language models (LLMs) have made significant strides in natural language processing (NLP) tasks (Brown et al., 2020). Recent work has shown that explicitly generating intermediate steps during reasoning tasks significantly improves a model's performance and interpretability (Shwartz et al., 2020; Paul and Frank, 2021; Marasovic et al., 2022; Lampinen et al., 2022; Wei et al., 2022). Producing such intermediate representations provides insight into the model's predictions and allows humans to inspect the model's reasoning process. However, these intermediate representations[1] can be unreliable (Ye and Durrett, 2022) and result in poor performance on downstream reasoning tasks. Most importantly, it is unclear how to meaningfully refine the intermediate representations to further improve the final performance.

[1] In a reasoning task, the intermediate representations can be viewed as inference rules, explanations or reasoning steps.

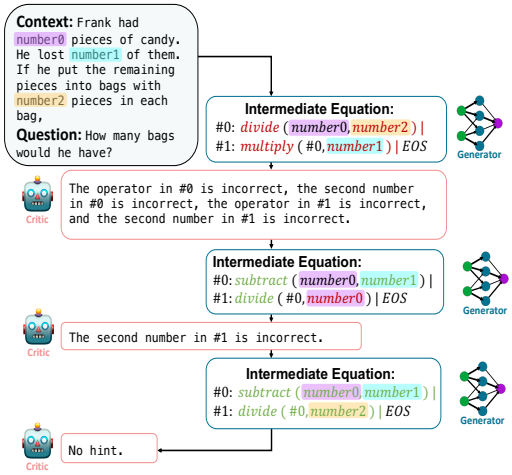

Figure 1: **REFINER example.** The critic model provides the generator model with feedback on its reasoning errors after evaluating the generated intermediate steps. The feedback, alongside the original question and previous intermediate equation, are fed back to the generator model.

The standard practice for correcting reasoning errors is to annotate new data and either retrain or finetune the model (Feng et al., 2021; Hedderich et al., 2021). However, fixing such errors by finetuning with more data is not only data- and resource-intensive but can also be insufficient to generalize well in complex reasoning tasks (Ward et al., 2022). Other works have explored improving models using feedback by providing a scalar reward (Ziegler et al., 2019; Martin et al., 2022) or directly revealing the correct missing answer (Mehta and Goldwasser, 2019; Elgohary et al., 2021; Tandon et al., 2022). However, in natural language reasoning tasks, defining a reward that captures different fine-grained reasoning error types (*e.g.,* semantic consistency, logical, etc.) remains an open challenge (Golovneva et al., 2023). Additionally, such

a reward provides a relatively sparse training signal.

In this work, we instead provide fine-grained and structured feedback on reasoning errors. We present REFINER, a novel interaction-based framework that allows a generator LM to iteratively use fine-grained feedback and refine its reasoning. The interaction happens between two models: a *generator*, which learns to solve the task by first generating the intermediate reasoning steps, and a *critic*, which provides structured feedback to the generator about errors in the intermediate steps.

To provide fine-grained feedback about reasoning errors, we develop a scheme to independently train the critic model on automatically constructed feedback data. More specifically, we create pairs of incorrect intermediate representations and structured[2] feedback on their fine-grained reasoning errors. Then, we use this data to train the critic to provide fine-grained feedback on erroneous intermediate reasoning steps. Finally, the critic interacts with the generator LM, offering feedback both during the training of the generator and during inference.

Figure 1 illustrates an example of our REFINER framework where, given a math word problem, the generator generates an equation as an intermediate representation. The critic identifies the errors in the equation and provides semi-structured textual feedback (*e.g.,* `"the operator in #0 is incorrect"`) to the generator. By interacting with the critic, REFINER enables the generator to reason over the semi-structured feedback and *refine* its generation.

Contributions. (i) We propose REFINER, a framework that refines LMs reasoning capabilities through feedback. Our work investigates how interacting with fine-grained reasoning feedback on intermediate reasoning steps impacts the performance of LMs on reasoning tasks. We evaluate REFINER on three natural language reasoning tasks: math word problems, synthetic natural language reasoning, and moral action generation. REFINER demonstrates significant performance gains across different LM architectures with different scales. Across different reasoning tasks, REFINER outperforms comparably-sized strong fine-tuned LM baselines (by +13.1, +3.2, +15 pts., respectively). (ii) We empirically demonstrate that for math word problems and synthetic natural language reasoning, our trained critic models alone are beneficial for improving intermediate representations as they help GPT-3.5 significantly increase its performance in a few-shot setting (by +3.5, +6.8 pts., respectively). We also demonstrate that providing structured feedback on fine-grained errors can benefit more than scalar value feedback for moral action generation and math word problem tasks. Our critic model acts as a 'reasoning refinement tool' for LLMs. (iii) We show that REFINER can substantially outperform other refinement methods that use feedback from large LMs, such as self-refine. (iv) Our analyses illustrate that (a) improving the intermediate representation generation improves the performance on the reasoning tasks, and (b) training a generator with an imperfect (noisy) critic is still beneficial. Our code and data will be made publicly available.

[2]Note that we transform the structured feedback into semi-structured textual feedback using templates.

2 Related Work

Intermediate Representations. While state-of-the-art LMs achieve incredible performances in a wide range of tasks, they have difficulty with many reasoning tasks (Wang et al., 2022), especially ones with multiple constraints or sub-problems or requiring specialized knowledge (Austin et al., 2021) – such as mathematical problem solving (Ling et al., 2017; Andor et al., 2019; Ran et al., 2019; Geva et al., 2020; Piękos et al., 2021; Cobbe et al., 2021a; Kim et al., 2022).

For these tasks, both intermediate representations and rationales have been shown to be beneficial in learning mathematical skills (Piękos et al., 2021), intermediate program execution computations (Nye et al., 2021), or general reasoning outputs (Wei et al., 2022; Golovneva et al., 2022).

Our work builds upon the observation that generating intermediate steps are valuable but distinguishes itself in several key aspects. Firstly, instead of prompting a large model, we finetune smaller models to learn to generate intermediate steps. Secondly, our framework can accommodate tasks that do not necessarily have unique closed-form correct answer, such as the *Moral Norm* task (see §3). Finally, our framework is trained with a critic providing feedback, improving the model's reasoning process and teaching it how to leverage feedback.

Natural Language Feedback. Recent work has explored giving models richer and more complex feedback through the use of natural language (Ziegler et al., 2019; Nguyen et al., 2021; Scheurer et al., 2022), used for aligning LLMs' output with

users' preferences (Christiano et al., 2017; Ziegler et al., 2019; Saunders et al., 2022; Scheurer et al., 2022; Bai et al., 2022), or to directly improve the model's performance in its current task (Weston, 2016; Rupprecht et al., 2018; Elgohary et al., 2020; Austin et al., 2021; Madaan et al., 2023). This training depends on human-created feedback, generated in large quantities (Bai et al., 2022), which takes up considerable resources. Though an external feedback provider can guide models to correct answers and reasoning (Austin et al., 2021), demonstrably better than they can themselves (Saunders et al., 2022), feedback has rarely been used in this way – and automated critics for reasoning tasks have proved to be difficult (Scheurer et al., 2022; Wang et al., 2022; Huang et al., 2022).

Recently, Welleck et al. (2022) introduced a secondary model, the corrector, which improves the initial proposition of a generation model, by learning the kind of mistakes made by the generator and how to fix them. In this work, we also use a secondary model, a critic, but apply it quite differently as we integrate it into an interaction loop with the generator model during training. We further differ from previous works as we provide feedback at the intermediate reasoning steps of the model and not at the final output. The feedback is thus closer to the source of mistakes and guides the model's reasoning toward the correct answer. Additionally, intermediate steps are often structured, allowing the critic to provide precise feedback.

3 REFINER

Problem Formulation. In this paper, we view *natural language reasoning* (NLR) as an autoregressive generation task where, given input context x, a model needs to generate y, such that y satisfies the constraints of the task. Usually, to generate correct or plausible y, the model needs to make the correct inference z as intermediate steps.[3] We decompose NLR tasks as follows: $p(y|x) = p(y|x,z)p(z|x)$. In practice, one can compute each conditional using an LM that includes its conditioning variables as a part of its input.

Before continuing with the model description, we describe three NLR tasks where we conduct our study and their respective intermediate representation z. We deliberately chose these three tasks since they broadly cover two types of reasoning: (i) logical reasoning and (ii) normative reasoning. They are exemplified in Appx Fig. 6 and detailed below.

Math word problem (MWP), where given a word problem x consisting of a context and question, the goal is to map x to a valid mathematical expression z (the intermediate representation) and then to a solution y. This task requires the model to perform deduction using mathematical reasoning.
Synthetic natural language reasoning (sNLR), where given a reasoning scenario x consisting of 5 synthetic rules and a fact, the model needs to deduce a conclusion y. This task requires the model to perform deductive reasoning and generate intermediate steps z and the conclusion y using closed-world rules and facts.
Moral norm and action generation for moral stories (MS), where given a context x consisting of a *situation*, an *intention*, and an *immoral action*, the model needs to generate the moral norm z and the moral action y. Moral actions are encouraged by the moral norm. This task requires the model to perform abductive reasoning to generate moral norms and deductive reasoning for moral action.

We propose to solve these tasks by forcing the model to generate intermediate hypotheses (z) and improving them via structured feedback. We introduce an interactive framework, REFINER, made of two separate models: (a) a CRITIC model (§3.1) trained to provide structured feedback on intermediate reasoning steps and (b) a GENERATOR model trained to solve the reasoning task by first generating intermediate reasoning steps (§3.2). The core idea of REFINER is to exploit the interaction between the generator model and the critic model, where the generator's intermediate reasoning steps are improved via structured feedback from the critic.

REFINER presents several important properties. First, the generator is trained to incorporate and leverage feedback, which helps it converge towards better reasoning during training and makes it capable of integrating feedback at test time, whether from a trained critic or a human (see §5). Second, the trained critic can be useful on its own; we demonstrate that a generalist LLM like GPT-3.5 can significantly benefit from interacting with our trained critic on the reasoning tasks we consider (see §5). Finally, having two separate models allows us to easily measure the benefits of feedback during training and/or during inference (see §6).

[3] We use "inference steps/representations" and "hypothesis" interchangeably.

Tasks	Error Types	Feedbacks
MWP	Incorrect Numbers	The `position` number in `equation-number` is incorrect.
	Incorrect Operators	The operator in `equation-number` is incorrect.
	Missing Operators	An operator is missing.
sNLR	Logically Invalid	The `X operator` makes `inference rule number` invalid.
	Missing Link	Missing link between the fact the rules.
	Missing Implicit Knowledge Step	The implicit knowledge is missing.
MS	Contradiction	Contradiction
	Semantic Misalignment	Semantically misaligned: " text snippet"

Table 1: An overview of the Error Types and Feedbacks for each reasoning tasks.

3.1 CRITIC Model

The role of the critic is to provide feedback on the intermediate hypotheses produced by the generator model. One way to evaluate the quality of the hypothesis and produce feedback on the hypothesis z, would be to compare it against a gold hypothesis z^*. Previous works employed automatic metrics like BLEU, ROUGE, etc., as value functions (Wu et al., 2018; Ramamurthy et al., 2022). However, these scalar value functions are not suitable for natural language reasoning tasks because (i) it is unclear how to define a scalar value function that can encapsulate fine-grained reasoning errors (Golovneva et al., 2023) and (ii) during inference, these functions require access to the gold hypothesis (which is unavailable in practice). Therefore, we train a critic model and endow it with the ability to evaluate the hypothesis in a fine-grained manner and provide structured feedback.

Feedback Data Generation. To train the critic, we have to create example pairs of implausible hypotheses and their corresponding feedback with fine-grained reasoning errors. Inspired by Golovneva et al. (2023) and Talmor et al. (2020), we first define fine-grained reasoning error types for each reasoning task (see Table 1). For MWP, an equation can be incorrect due to: (i) the operands or operators in the equations being incorrect and/or (ii) one or more operators missing. For sNLR, an inference rule can be incorrect because it is (i) logically invalid and/or (ii) missing reasoning rules (*failing to connect the correct facts with correct rules or missing implicit knowledge*). For MS, a moral norm can be incorrect due to (i) contradiction and/or (ii) semantic misalignment.

Based on these error types, we propose two strategies to create the feedback data: (i) **Rule-based perturbation** strategy: we perturb the plausible hypotheses (z) in the training data and collect

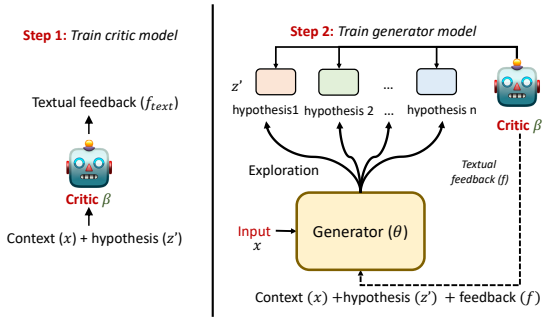

Figure 2: Overview of REFINER interaction loop. Left side: Training the critic model. Right side: In each iteration, the generator generates multiple hypotheses. The critic randomly selects one hypothesis and provides feedback based on reasoning errors.

a pool of data D (x: input, z: plausible hypothesis, z': implausible hypothesis). We perturb by omitting, replacing or adding some tokens or some rules from the plausible hypothesis to create an implausible hypothesis automatically (details in Appendix F.1). (ii) **Synthetic Generation** strategy: we prompted OpenAI's GPT-3.5 to generate implausible hypotheses based on the error types automatically. We used a few-shot setting where we varied the instruction, the number of demonstrations, and the formatting of the demonstrations (details in Appendix F.2).

Since our perturbations and automatic implausible hypotheses are based on logic and reasoning errors, we create structured feedback f for every example (x, z, z') by stating the error type that occurs in z' but not in z (see Table 1). The basic structure of feedback f for these tasks is $\langle$*error type, position (optional), hint (optional)*$\rangle$, where position denotes the error position in the implausible hypothesis (see Table 1). Despite the simplicity of the strategy we used for our tasks, this approach is easily generalisable to other reasoning tasks.

We also replace the correct judgment with random judgments to scale the number of implausible hypotheses per example. Finally, as feedback f, we provide <*error type, hint*>. For non-monotonic reasoning tasks like norm and action generation, the critic should be able to provide hints that align the generator model's objective to the reasoning task. Hence, as a *hint*, we provide verb phrases from the norms. Since the critic provides textual feedback to the generator, we convert the structured feedback into natural language feedback [4]. Formally, we create a data pool $D = \{x, z, z', f\}$ to train a critic model.

[4]Further details about feedback are provided in Appx.F.

Training the critic model. We train a supervised CRITIC model (π_β) with the context (x) and (plausible or implausible) hypothesis (z or z') as input and the textual feedback as output. We update the CRITIC with the cross-entropy loss: $L(\beta) = -\log p_\beta(f(u)|x,u)$ where $u \in z, z'$. The trained critic is only used during inference. The oracle critic is used while training the generator.

3.2 GENERATOR Model

This section presents a generator model that iteratively learns to interact with the CRITIC model.

Warm-up. Given a context x the generator model (π_θ) is trained to generate plausible hypotheses. The warm-up phase is critical to ensure that, when the critic comes in the loop, the generator does not produce random answers likely to be bad, given the size of the output space. As such, we use a small supervised dataset (10% training data) to fine-tune the model on the NLR task of interest. After the warm-up phase, we use the additional feedback f from the critic model and learn $\pi_\theta(z|x,z',f)$.

Exploration. At each iteration (t), the generator model generates multiple hypotheses (z^k) using nucleus sampling. The critic model randomly selects one hypothesis and provides feedback on that hypothesis. The exploration step aims at increasing the output variance such that the generator receives a wide range of feedback during training.

Learning. We update the GENERATOR model using the following cross-entropy loss: $L(\theta) = -\sum_{t=1}^{T} \log p_\theta(z_t|x,z'_t,f_t(z'))$ where T = total number of iterations. Since the feedback contains the error types and hints, which are (latent) fine-grained and logical, it should allow the model to learn and update its generation by addressing the reasoning errors mentioned in the feedback.

Inference. We use the trained critic along with the trained generator to generate a trajectory $z_0, z_1, ..., z_T$ and stop when either $f(z_t)$ is generated by the generator or *"No hint"* is generated by the critic. We also experimented with *chain of thought* prompting, where the generator generates a trajectory $z_0y_0, z_1y_1, ..., z_Ty_T$ and stops when the critic generates *"No hint"*.

4 Experimental Setup

Datasets. We evaluate REFINER on three diverse tasks (examples in Fig. 6). We briefly describe the datasets used for each task below.*Math Word Problem* (MWP): We train our models on MAWPs (Koncel-Kedziorski et al., 2016) dataset and evaluated our models on a challenging dataset SVAMP (Patel et al., 2021). We evaluate our model on both the equation generation (z) and answer prediction (y) tasks. Similar to Ling et al. (2017); Amini et al. (2019) for equation generation, we replace the numeric values with variable names, for example, `number0`, `number1`, etc. Further, we also evaluated on GSM8K (Cobbe et al., 2021b) dataset which consists of 8.5K high-quality linguistically diverse grade school math word problems. For *Synthetic Natural Language Reasoning* (sNLR), we use the dataset from Liang et al. (2022) with the difficulty level as hard. We evaluate our model on both inference rule generation (z) and consequent generation (y). For *Moral Story* (MS), we use a dataset from (Emelin et al., 2021), where we evaluate our model on moral norm z and the moral action y generation.

Model	Eq. (z)	Ans. (y)
UQA-base	34.1	–
UQA-base + PPO	31.5	–
REFINER $_{base}$	**47.2**	–
UQA-large	46.7	–
UQA-large + PPO	48.2	–
REFINER $_{large}$	**53.8**	–
GPT-3.5 + CoT	64.1	67.1
GPT-3.5 + CoT + REFINER$_{critic}$	**67.3**	**70.6**

Table 2: Results on MWP. Comparison of REFINER with baselines on the SVAMP dataset. The average score over three runs is reported (p<0.05). For models other than GPT-3.5, the answer can be obtained via symbolic execution of the equation and is thus a function of the validity of the equation.

Training Details. For each task, we train a UnifiedQA-T5-base model (UQA-base) (Khashabi et al., 2020) as a critic (§3.1). Further training and inference details are provided in Appendix D. On the MWP and sNLR tasks, we use the exact match (EM) metric for intermediate steps (equation generation and inference rules) and accuracy (Acc) for the final answers. For MS, we conduct a manual evaluation study to assess the relevance of norms and moral actions [5]. To train the critic model, we used the feedback data generated using the rule-based perturbation strategy (see §3.1).

Baselines. We compare our method with three different LMs as generator models: UQA-base, UQA-large (supervised setting), GPT-3.5-

[5]We conduct a manual evaluation since automatic scores such as BLUE, ROUGE, etc. only account for the word-level similarity between the gold and the generated norms or actions. Further evaluation details are provided in Appendix G.

Model	IR (z)	Con (y)
UQA-base	90.6 ± 0.8	94.1
REFINER $_{base}$	**93.5** ± **0.4**	**97.3**
GPT-3.5 + CoT	14.3 ± 0.9	40.6
GPT-3.5 + CoT + REFINER	**21.1** ± **1.2**	**42.1**

Table 3: Results on sNLR task. The average score over three runs is reported (p<0.05). IR: Inference Rules (Exact Match), Con: Consequent (Accuracy)

Model	Norm (z)				Action (y)			
	I↓	U↓	R↑	α	I↓	U↓	R↑	α
B	34	17	49	0.35	28	14	58	0.64
B+PPO	38	10	52	0.38	31	17	52	0.38
REFINER	19	12	**69**	0.33	18	9	**73**	0.55

Table 4: Results on Moral Norm and Moral Action. We report human evaluation. B: UQA-large; I: *Irrelevant*, U: *Unsure*; R: *Relevant*; α: Krippendorff's alpha

`text-DaVinci-003` and ChatGPT (few-shot setting). We also compare REFINER to *Proximal Policy Optimization* (PPO) RL-based method (Schulman et al., 2017). We use the implementation of PPO from (Ramamurthy et al., 2022). For GPT-3.5, we provide 2 for demonstrations per class. We also experimented with *chain of thought* (COT) prompting (Wei et al., 2022) where the model is prompted first to generate the intermediate steps (z) and then the final answer (y). Note that the sNLR task is a synthetic task where the model needs to perform either one-hop or two-hop reasoning. Clark et al. (2021) showed that fine-tuning large language models (354M parameter size) could achieve (99% accuracy) high performance. Hence, we only compare our REFINER model with the UQA-base model (220M) (see Table 3). Since human annotation is expensive, we focus on comparing against the most meaningful baseline: UQA-large for MS task (see Table 4). It is important to highlight that our proposed framework is general, and one can use any other LMs as GENERATOR or CRITIC.

5 Results

We evaluate our model on two aspects (i) performance on intermediate steps and (ii) performance on the final answer prediction. Tables 2, 3, and 4 show the performance comparisons.

Performance on Intermediate Steps. Table 2 reports the performance of the MWP task. We explored two different scenarios: (i) where the model only generates the equations (z) with variable names replacing the numeric values, and (ii) where the model generates both the equations and the final answers together. We observe for both scenarios that REFINER significantly outperforms baseline models with comparable sizes. Notably, UQA-base benefits most (+13.1 EM) when adding a critic in the loop. We observe that GPT-3.5 significantly benefits from the REFINER trained critic. Since LLMs like GPT-3.5 (175B parameters) are expensive to finetune, the improvement in equation generation of +3.2 EM without any modification is important. Interestingly, we observe that GPT-3.5 + COT manages to have significantly higher accuracy in answer y than in equation z (see Table 2). This result is similar to the observation made by Ye and Durrett (2022) and suggests that the intermediate equations can be unreliable. Finally, REFINER could even outperform PPO, which uses BLEU-score as a reward function. This suggests that semi-structured fine-grained textual feedback is more beneficial than value-based (where values are from automatic metrics) reward feedback. Note that this result may vary when these models are optimized directly with complex human values, as shown in Stiennon et al. (2020). Qualitatively, REFINER can correct incorrect equations through structured feedback, fixing the operators within a multistep solution (see Fig. 7).

For sNLR, similar to Liang et al. (2022), we observe that GPT-3.5 performs poorly (see Table 3). REFINER improves +2.9, and +6.8 EM scores over UQA-base, and GPT-3.5, respectively. Contrary to the MWP, the final answer y is not a symbolic execution away from the intermediate step z, but we still observe that REFINER focuses on improving the intermediate step z, resulting in significant improvements in the answer y prediction. Again, we observe that REFINER with a UQA-base can outperform few-shot prompted GPT-3.5. Thus, our critic can identify the fine-grained reasoning errors and help improve the performance on inference rules generation.

For MS, we assess the generation quality with three human judges who indicate whether the generated norms and moral actions are relevant to the given moral story. Table 4 summarises human evaluation results on 100 moral story examples randomly sampled from the MS test dataset. More specifically, we report evaluation breakdown for both norm and moral action by the number of instances that are either *Irrelevant*, *Unsure* or *Relevant* along with Krippendorf's α (Krippendorff, 2018) agreement scores. The results show an improvement of 20 points, increasing the relevance over a strong UQA-large baseline. Hence, this sug-

Model	SVAMP		GSM8K	
	GPT-3.5	ChatGPT	GPT-3.5	ChatGPT
CoT	67.1	68.2	63.5	74.1
Self-reflection	67.2	68.4	63.1	74.6
Self-refine	67.6	68.2	63.8	74.7
REFINER	**70.6**	**71.4**	**66.2**	**75.9**
ReACT	67.3	68.4	64.7	75.5
ReACT + REFINER	**70.6**	**71.9**	**67.8**	**77.4**
Self-consistency	69.5	70.4	65.5	76.1
Self-consistency + REFINER	**72.1**	**72.5**	**67.2**	**78.1**

Table 5: **Comparison with different refinement methods** on SVAMP and GSM8K datasets. Averaged accuracy over three runs on the test sets is reported (p<0.05).

gests that a specialized critic model with 3 times fewer parameters than the generator can improve the performance on generating reasoning steps.

Performance on Final Answer Prediction. We observe that REFINER outperforms the strong LM baselines by $+3.5, +3.2, +15$ points for MWP, sNLR, and MS, respectively. These results support our hypothesis that generating better intermediate steps can result in better answer prediction. Notably, on the sNLR task, for GPT-3.5, we observe that by adding a critic, there is an improvement of +6.8 in inference step generation; however, only $+1.5$ in the consequent prediction. This result indicates that LLMs may either not use these intermediate steps to perform the deduction or fail to perform deduction.

Comparing REFINER with other refinement methods. In Table 5, we compare REFINER with two other recent refinement methods: Self-refine (Madaan et al., 2023) and Self-reflection (Shinn et al., 2023) method on the SVAMP and GSM8K datasets. Both these baseline methods use LLMs to generate automatic feedback. Similar to Madaan et al. (2023), we observe that self-refine has minor improvement for MWP tasks. On the contrary, we find that REFINER significantly improves the performance of GPT-3.5 and ChatGPT by +3.3 and +2.2 on SVAMP and GSM8K datasets, respectively. This highlights the benefit of training a *specialised critic* that is grounded to the task. It can make LLMs more accurate than feedback from a general-purpose model (GPT-3.5 or ChatGPT). In Appendix §A.1, we have provided more details about the quality of feedback generated using our trained critic and GPT-3.5 (see Table 7). Further, we assess the performance of REFINER in improving the CoT generated by two recent methods: Self-Consistency (Wang et al., 2023) and ReACT method (Yao et al., 2023). We observe that REFINER can improve self-consistency and ReACT

Model	Eq. (z)
REFINER $_{base}$ + critic data$_{rule-based}$	47.2
REFINER $_{base}$ - critic$_{inference}$	39.8
REFINER $_{base}$ - critic$_{inference}$ - exp	37.4
REFINER $_{base}$ - critic$_{training}$	34.1
REFINER $_{base}$ + critic data$_{synthetic}$	44.1
REFINER $_{base}$ + critic$_{Oracle}$	66.0

Table 6: **Ablation Result** on MWP task; Comparing model without critic during inference, and without the exploration (exp) phase during training. We report the exact match scores of the generated equation, comparable to Table 2.

by +2.02 and +2.9. This demonstrates that a trained critic can be used as a *tool* and can bring performance gains to different methods out-of-the-box (more details in Appendix §A.2).

Ablation. To obtain better insight into the contributions of the individual components of our models, we perform an ablation study (Table 6). We observe that there is a considerable drop in performance from 47.2 to 39.8 when we do not use the critic model during inference. Hence, this result indicates that our generator model can leverage the feedback from the critic at inference time. Further, we find that the exploration step improves the performance $+3.3$ over the baseline model. This result supports our hypothesis that the exploration step increases the output variance and gives the generator model the opportunity to learn over a wide range of feedback. We compared the performance with the critic model trained on two different training data (see §3.1). We find that the critic trained on small automatically generated data using GPT-3.5 works better than without the critic in the loop. This result motivates researchers to use this method to generate negative samples to train their critic or preference learning model. Finally, we also observe that if the critic was perfect (Oracle), then REFINER can significantly improve the performance by fixing the mistakes generated by the generator model. This result indicates that REFINER can be seen as a framework that allows AI-AI and human-AI interaction.

6 Analysis

Error Analysis. In order to get more insight into the performance of our method, we conduct a fine-grained error analysis on the MWP and MS datasets (Fig. 3). We note that the most frequent errors are *Incorrect Numbers* for MWP and *Semantic Misalignment* for MS. An intuitive reason can be that for the MWP task, the models are sensitive to the

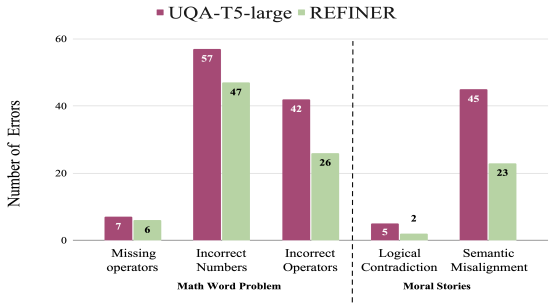

Figure 3: **Error analysis.** Number of errors made by baseline UQA-large and REFINER on 100 instances sampled randomly from test sets of both datasets. Errors are categorized according to Table 1).

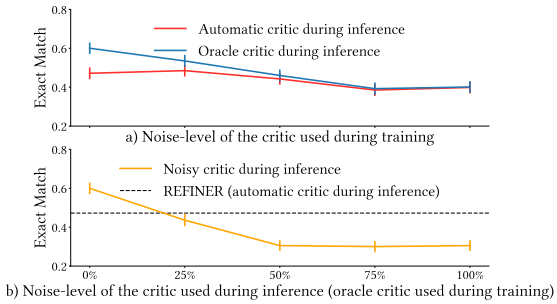

Figure 4: **Noisy-critics analysis**. In plot (a), we vary the noise level of the critic used during training (0 noise corresponds to oracle) and compare the resulting models when using the oracle and the training automatic critic during inference. In plot (b), we train with the oracle critic but vary the noise level of the critic used during inference.

numbers order as argued in (Patel et al., 2021). For MS, generating norms grounded in the context is challenging. Our analyses show a clear trend that REFINER is able to considerably reduce the errors for both datasets. This indicates that our trained critic model could identify fine-grained reasoning errors during inference.

Noise Sensitivity. To further understand the behaviour of the REFINER framework, we run variations with noisy critics for the MWP task. We replace the oracle critic used during training with a noisy critic in (Fig. 4 (a)) to inspect how training with an imperfect critic impacts the generator. We also use a noisy critic at inference while keep the oracle critic during training (in Fig. 4 (b)). The noisy critics are generated by random perturbations of the oracle critic; for a noise-level ϵ, the oracle feedback is replaced by random feedback with probability ϵ.

Fig. 4 (a) shows that when training with a very noisy critic ($>75\%$ noise), the generator LM learns to ignore the critic, as there is no difference between using the trained critic or the oracle during inference. Interestingly, training with a bit of noise ($<50\%$) does not seem to harm the model, as performances are not statistically different than training with the oracle (noise of 0%). Fig. 4 (b) depicts the quality of the critic used at inference time has a huge impact. Having oracle provide feedback is by far the best scenario. Already with 25% noise, the critic makes the generator perform worse than using our trained critic (REFINER). With more than 50% noise, the critic significantly harms the generator. The generator, trained with an oracle critic, has learned to trust the critic and expects useful feedback.

Qualitative Analysis. To explain the findings in §6, we further manually analyze 100 instances for the MWP task. We observe two different scenarios when REFINER failed to fix the outputs generated by GENERATOR model: (a) when the CRITIC model provides a *correct* feedback; however, the GENERATOR model still generates *incorrect* equation, and (b) the CRITIC model provides an *incomplete* or *partially correct* feedback. The former case indicates that either the GENERATOR model makes mistakes in following the instruction from the CRITIC or the feedback from the critic can be ambiguous. For example, in Appx Fig. 5, (b) we observe the case when the critic is correct, but the feedback could result in an incorrect equation. The latter case indicates that our trained critic model generates incorrect feedback, which can result in incorrect or partially correct equations. We also observe that our CRITIC model failed to generate correct feedback when the GENERATOR model generates incorrect equations with multiple mistakes.

7 Conclusion

In this paper, we propose REFINER, a framework to improve the reasoning abilities of LMs through an iterative feedback loop between two models, a *generator* and a *critic*. Our evaluation of this framework on three reasoning tasks showed structured and fine-grained feedback on intermediate reasoning errors results in significant performance gains, surpassing scalar value feedback. Our trained critic model alone, even when noisy, can improve intermediate representations of LMs, showing that REFINER can significantly boost LMs' performance on reasoning tasks. Our REFINER framework is very general and, in principle, might be applied to steer language models in performing different reasoning tasks. More specifically, the *critic* model can be seen as a tool for LLMs to refine their generation quality.

Limitations

Our REFINER framework could not be comprehensively evaluated on all applicable downstream reasoning tasks due to their sheer number. While deliberately distinct, we focused on only three different reasoning tasks in order to study how natural language reasoning feedback can impact downstream tasks. We believe this represents an initial but important step towards exploring automated natural language feedback on intermediate representations. In addition, the critic we presented here is specific for each task, while the ideal critic would be a general one, capable of providing feedback on a wide range of reasoning tasks. Similarly, we considered fine-grained reasoning errors specific to each reasoning task. Recent work has mentioned several other fine-grained reasoning errors (Golovneva et al., 2023), which can't be fully covered by the reasoning tasks we considered. Generalizing both the critic and fine-grained error types emerges as both the main limitations of this paper and the directions of future work. Finally, with LLMs being deployed more and more for real-life applications (medical domain, making important decisions), we believe it is crucial to develop expert models and automatic feedback mechanisms to inspect model generations and improve them. LLMs are impressive and work well on several NLP tasks, but they are not expert systems. Our work aims to address this gap by showing that adding interventions/feedback from critics (specialised finetuned critics) can help the LLM model to be more accurate—additionally, making the whole process more transparent.

Ethical Considerations

In this paper, we experiment with existing datasets which are, to the best of our knowledge, adequately cited. Our proposed framework REFINER is designed to improve the reasoning abilities of LMs. These LMs have been shown to encode biases about race, gender, and many other demographic attributes (Weidinger et al., 2021), (Sheng et al., 2020). Since our framework does not offer a way to mitigate these biases, models improved using this framework could still reflect the same harmful behaviours normally exhibited by these models. We recommend anyone deploying our model *off-the-shelf* should first check whether the model is harmful towards any protected group, and appropriate mitigation should be taken. In addition, our MS task is based on a dataset of situations, intentions, and actions that heavily skew towards Western culture and social norms (Emelin et al., 2021). Consequently, our human evaluation on the MS task was done with AMT workers based in the US who were paid adequately for the average time it took to solve the task.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving BERT a calculator: Finding operations and arguments with reading comprehension. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5947–5952, Hong Kong, China. Association for Computational Linguistics.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022. Constitutional ai: Harmlessness from ai feedback.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens

Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2021. Transformers as soft reasoners over language. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI'20.

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. 2021a. Training verifiers to solve math word problems.

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. 2021b. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

Ahmed Elgohary, Saghar Hosseini, and Ahmed Hassan Awadallah. 2020. Speak to your parser: Interactive text-to-SQL with natural language feedback. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2065–2077, Online. Association for Computational Linguistics.

Ahmed Elgohary, Christopher Meek, Matthew Richardson, Adam Fourney, Gonzalo Ramos, and Ahmed Hassan Awadallah. 2021. NL-EDIT: Correcting semantic parse errors through natural language interaction. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5599–5610, Online. Association for Computational Linguistics.

Denis Emelin, Ronan Le Bras, Jena D. Hwang, Maxwell Forbes, and Yejin Choi. 2021. Moral stories: Situated reasoning about norms, intents, actions, and their consequences. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 698–718, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 968–988, Online. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 946–958, Online. Association for Computational Linguistics.

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2022. Roscoe: A suite of metrics for scoring step-by-step reasoning.

Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. ROSCOE: A suite of metrics for scoring step-by-step reasoning. In The Eleventh International Conference on Learning Representations.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2545–2568, Online. Association for Computational Linguistics.

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

Zhanming Jie, Jierui Li, and Wei Lu. 2022. Learning to reason deductively: Math word problem solving as complex relation extraction. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5944–5955, Dublin, Ireland. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1896–1907, Online. Association for Computational Linguistics.

Niklas Kiehne, Hermann Kroll, and Wolf-Tilo Balke. 2022. Contextualizing language models for norms diverging from social majority. In Findings of the EMNLP 2022, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics, Association for Computational Linguistics.

Bugeun Kim, Kyung Seo Ki, Sangkyu Rhim, and Gahgene Gweon. 2022. EPT-X: An expression-pointer transformer model that generates eXplanations for numbers. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 4442–4458, Dublin, Ireland. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS:

A math word problem repository. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Klaus Krippendorff. 2018. Content analysis: An introduction to its methodology. Sage Publications.

Andrew K Lampinen, Nicholas A Roy, Ishita Dasgupta, Stephanie CY Chan, Allison C Tam, James L McClelland, Chen Yan, Adam Santoro, Neil C Rabinowitz, Jane X Wang, et al. 2022. Tell me why! explanations support learning relational and causal structure. In International Conference on Machine Learning.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. arXiv preprint arXiv:2211.09110.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

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

Ana Marasovic, Iz Beltagy, Doug Downey, and Matthew Peters. 2022. Few-shot self-rationalization with natural language prompts. In Findings of the Association for Computational Linguistics: NAACL 2022, pages 410–424, Seattle, United States. Association for Computational Linguistics.

Alice Martin, Guillaume Quispe, Charles Ollion, Sylvain Le Corff, Florian Strub, and Olivier Pietquin. 2022. Learning natural language generation with truncated reinforcement learning. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 12–37, Seattle, United States. Association for Computational Linguistics.

Nikhil Mehta and Dan Goldwasser. 2019. Improving natural language interaction with robots using advice. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1962–1967, Minneapolis, Minnesota. Association for Computational Linguistics.

Khanh Nguyen, Dipendra Misra, Robert Schapire, Miro Dudík, and Patrick Shafto. 2021. Interactive learning from activity description.

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. 2021. Show your work: Scratchpads for intermediate computation with language models.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2080–2094, Online. Association for Computational Linguistics.

Debjit Paul and Anette Frank. 2021. COINS: Dynamically generating COntextualized inference rules for narrative story completion. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 5086–5099, Online. Association for Computational Linguistics.

Piotr Piękos, Mateusz Malinowski, and Henryk Michalewski. 2021. Measuring and improving BERT's mathematical abilities by predicting the order of reasoning. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 383–394, Online. Association for Computational Linguistics.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. 2022. Is reinforcement learning (not) for natural language processing?: Benchmarks, baselines, and building blocks for natural language policy optimization.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. NumNet: Machine reading comprehension with numerical reasoning. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2474–2484, Hong Kong, China. Association for Computational Linguistics.

Christian Rupprecht, Iro Laina, Nassir Navab, Gregory D. Hager, and Federico Tombari. 2018. Guide me: Interacting with deep networks. CoRR, abs/1803.11544.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators.

Jérémy Scheurer, Jon Ander Campos, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2022. Training language models with language feedback.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. ArXiv, abs/1707.06347.

Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. 2020. Towards Controllable Biases in Language Generation. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 3239–3254, Online. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4615–4629, Online. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. In Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. In Advances in Neural Information Processing Systems, volume 33, pages 20227–20237. Curran Associates, Inc.

Niket Tandon, Aman Madaan, Peter Clark, and Yiming Yang. 2022. Learning to repair: Repairing model output errors after deployment using a dynamic memory of feedback. In Findings of the Association for Computational Linguistics: NAACL 2022, pages 339–352, Seattle, United States. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In The Eleventh International Conference on Learning Representations.

Francis Rhys Ward, Francesco Belardinelli, and Francesca Toni. 2022. Argumentative reward learning: Reasoning about human preferences. Workshop on Human-Machine Collaboration and Teaming at ICML, abs/2209.14010.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. CoRR, abs/2201.11903.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William S. Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2021. Ethical and social risks of harm from language models. CoRR, abs/2112.04359.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. Generating sequences by learning to self-correct.

Jason Weston. 2016. Dialog-based language learning.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45, Online. Association for Computational Linguistics.

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. A study of reinforcement learning for neural machine translation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3612–3621, Brussels, Belgium. Association for Computational Linguistics.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In International Joint Conference on Artificial Intelligence.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In The Eleventh International Conference on Learning Representations.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. In Advances in Neural Information Processing Systems.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3928–3937, Online. Association for Computational Linguistics.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences.

A Additional Results

A.1 Comparing trained vs training-free feedback.

To better understand the difference in the quality of the feedback, we compare our trained critic model with GPT-3.5 (see Table 7). Please note we also include instances where the critic feedback should say the solution is correct and hence generate 'No'. Our exact match metric is not order-sensitive. We extract the sentences and match them individually to the oracle answers. Since we focused only on the semi-structured critic feedback, automatic evaluation can already capture (measure effectively) the quality of the feedback. Here are the results on 500 instances per task to compare the model's performance in generating structured feedback. For GPT-3.5, we have provided (two) few-shot examples per type of error and two examples with 'No' as feedback. Our results show that trained critic (UQA) can comprehensively outperform GPT-3.5. The result suggests that GPT-3.5 is good at identifying when it is correct. It makes errors when asked to generate meaningful semi-structured feedback for incorrect reasoning steps. For Logical tasks, sNLR, GPT-3.5 performs poorly.

Task	UQA (220M)	UQA (770M)	GPT-3 (175B)
MWP	69.5 +/- 2.6	73.4 +/- 3.7	63.5 +/- 5.6
sNLR	95.5 +/- 1.4	98 +/- 2.2	34.5 +/- 2.4
MN	77.4 +/-2.5	80 +/- 4.5	76.4 +/-3.5

Table 7: **Comparison** trained feedback vs training-free method. The exact match score is reported.

A.2 Details about ReACT and Self-consistency

The ReACT method consists of the reason model (Reason-Only) LLM (GPT-3.5), which generates a single thought at each step, and the Action model LLM (another GPT-3.5) does the calculation and generates the intermediate outputs (observations). We propose to refine the intermediate steps generated by the above steps and report the results below. Please note ReAct is approx **3-4** times more expensive than GPT-3.5 + CoT. In our experiments, we assumed 3 reasoning steps for ReACT and a sample size of 5 for self-consistency to be more cost-effective. Interestingly, we observe that ReACT perform similarly to CoT for the SVAMP dataset. One intuitive reason is that the SVAMP dataset contains questions which require one or two-hop reasoning only.

A.3 More results on SVAMP dataset

In the MWP, for the answer prediction task, we compare REFINER with the previously reported baselines from Jie et al. (2022) including Graph2Tree (Zhang et al., 2020) that uses quantity relations using GCN; GTS (Xie and Sun, 2019) which is a sequence-to-tree model that mainly uses a tree-based decoder with GRU; and DeductReasoner (Jie et al., 2022) which uses bottom-up DAG-structured decoding. Results of this comparison can be found in Table 8. For the sNLR task, we also experiment with a critic model trained on 50% of its original training data and we still observe a performance improvement over the baseline as can be seen in Table 13.

Answer Prediction (y)	**Acc %**
GTS	30.8
Graph2Tree	36.5
BERT-Tree	32.4
Roberta-large-GTS	41.0
Roberta-large-Graph2Tree	43.8
Roberta-large-DeductReasoner	45.0
Few-Shot GPT-3	63.05
Few-Shot GPT-3 + COT	63.5
Few-Shot GPT-3 + COT + REFINER	**66.4**

Table 8: Results on SVAMP dataset

B REFINER Framework

Alg. 1 and Alg. 2 outline the training and inference algorithms for REFINER. We train a supervised CRITIC model (π_β) with the context (x) and (plausible or implausible) hypothesis (z or z') as input and the textual feedback as output. Given a context x the generator model (π_θ) is trained to generate plausible hypotheses.

Algorithm 1 REFINER Training

for E epochs **do**
  for $i(batch) \leftarrow 1$ to N **do**
    Initialize (feedback) $f_0 \leftarrow No$
    for $t \leftarrow 1$ to T **do**
      $\hat{z}^k_{i,t} \sim \pi_\theta(y_i|c_i, f_{t-1}, \hat{z}_{i,t-1})$
      $f_t, \hat{z} \leftarrow \pi_\beta(c_i, z_i, \hat{z}^k_{i,t})$
      $\mathscr{L}^{lm}_i$ += $-\log p(z_i|c_i, f_{t-1}, \hat{z}_{i,t-1})$
    end for
  end for
end for
return π_θ

Algorithm 2 REFINER Inference

Initialize *answers* $\leftarrow$ empty list
for $i(batch) \leftarrow 1$ to N **do**
  Initialize (reward) $r_i \leftarrow 0$, $p_i \leftarrow 1$
  Initialize (hint) $h_0, \hat{y}_{i,0} \leftarrow No, []$
  for (turn) $t \leftarrow 1$ to T **do**
    $\hat{y} \leftarrow \pi_\theta(y_i|c_i, h_{t-1}, \hat{y}_{i,t-1})$
    $h_t \leftarrow \pi_\beta(c_i, \hat{y}_i)$
    if h_t == No **then**
      answers.append($\hat{y}$)
      break
    end if
  end for
  answers.append($\hat{y}$)
end for
return *answers*

C Datasets and Models

In Table 9 and Table 11, we report the data statistics and dataset details. In Table 10, we report the details of the used models. Our research is conducted solely on datasets that are in the English language.

Task	Train	Dev	Test
MWP	3,138	–	1000
sNLR	1000	5000	5000
MS	10000	1000	1000
GSM8k	–	–	1319

Table 9: Dataset Statistics: nb. of instances.

Model	Parameter Size
UQA-base	220M
REFINER$_{base}$	440M
UQA-large	770M
REFINER$_{large}$	990M
GPT3.5	175B

Table 10: Model Sizes.

D Training Details

Training Details. For each task, we train a UnifiedQa-T5-base model (UQA-base) (Khashabi et al., 2020) as a critic (§3.1). Further evaluation details are provided in Appendix G. For exploration (§3.2), we use nucleus sampling with $p = 0.5$. We select the hyper-parameters by the validation loss: for both the generator and critic model, we use the Adam optimizer with a learning rate of $1e^{-4}$. Each model is trained for 20 epochs with early stopping based on validation loss. We trained all models on one A100 GPU. We run our models with 3 random seeds and report the average results. We perform a binomial sign test. We find that p-values are always <0.05 when we compare REFINER with all the baselines (GPT-3.5, Self-refine, Self-reflection), suggesting our results are not random and significant. For the human study, we selected outputs from the best models (baselines and our model) according to automatic metrics. We train models with $T = 3$ iterations. We trained the critic model for 8 hours and trained the generator model for 12 hours.

At inference time, we use greedy decoding for the generator and critic model with $T = 1$ for the automatic critic and $T = 3$ for the oracle critic. We evaluate our methods using the metrics presented in the original papers that proposed the tasks. On the MWP and sNLR tasks, we use the exact match (EM) metric for intermediate steps (equation generation and inference rules) and accuracy (Acc) for the final answers. For MS, we conduct a manual evaluation study to assess the relevance of norms and moral actions.[6]

E Qualitative Examples

Figure 7 and 19 depict a qualitative example of REFINER where REFINER could correct incorrect equations through structured feedback, fixing the operators within a multistep solution. Table 19 shows some qualitatively improved examples for MS.

F Feedback Data Generation

F.1 Rule-based Perturbation

Based on these error types, we perturb the plausible hypotheses (z) in the training data and collect a pool of data D (x: input, z: plausible hypothesis, z': implausible hypothesis). We perturb by omitting, replacing or adding some tokens or some rules from the plausible hypothesis to automatically create an implausible hypothesis. For example, in Fig. 6, for sNLR we omit a few inference steps from the correct hypothesis "`#0: viridian is green, #1: rose is green`" and create an incorrect (incomplete) hypothesis (see Fig. 6). Since our perturbations are based on logic and reasoning errors, we create structured feedback f for every

[6] Since the automatic scores such as BLUE, ROUGE, etc. only account for word level similarity between gold norms or actions and generate norms or actions.

Dataset/Tools	Citation	Link	License
SVAMP	Patel et al. (2021)	https://github.com/arkilpatel/SVAMP	MIT License
GSM8k	Cobbe et al. (2021b)	https://github.com/openai/grade-school-math	MIT License
sNLR	Liang et al. (2022)	https://github.com/stanford-crfm/helm	Apache License
Moral Norm	Emelin et al. (2021)	https://github.com/demelin/moral_stories	MIT License
HuggingFace	Wolf et al. (2020)	https://github.com/huggingface/transformers	Apache License

Table 11: More details about datasets and Tools

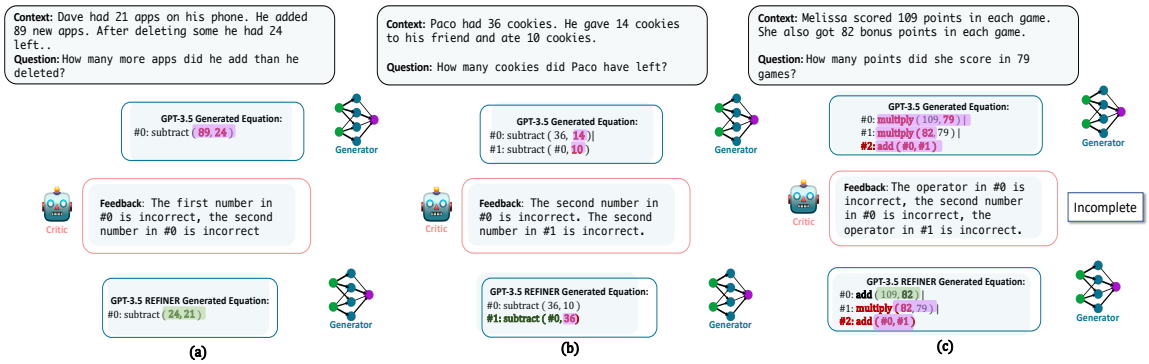

Figure 5: **Examples.** REFINER on MWP task. There are different scenarios are highlighted in the figure, where (a) the CRITIC model provides correct feedback, GENERATOR model utilizes the feedback and fixes the incorrect equation, (b) the CRITIC model provides a *correct* feedback however, GENERATOR model fails to fix the *incorrect* equation, and (c) the CRITIC model provides an *incomplete* feedback GENERATOR model partially fixes the incorrect equation.

Model	Eq. (z)	Ans. (y)
UQA-large	46.7	–
UQA-large + PPO	48.2	–
REFINER $_{large}$	**53.8**	–
REFINER $_{large}$ + Oracle (T=3)	68.1	–
GPT-3.5 + CoT	59.3	63.5
GPT-3.5 + CoT + REFINER$_{critic}$	62.3	**66.4**
GPT-3.5* + CoT	64.1	67.1
GPT-3.5* + CoT + REFINER$_{critic}$	**67.3**	**70.6**

Table 12: Results on MWP. Eq.: Equation, Ans. Answer. Comparison of REFINER with baselines on the SVAMP dataset. GPT-3.5: code-DaVinci-002, GPT-3.5*: text-DaVinci-002 For models other than GPT3.5, the answer can be obtained via symbolic execution of the equation and is thus a function of the validity of the equation. For GPT3.5, the model is few-shot prompted to either generate the equation with variable names z, or generate the answer y.

Model	IR	C
50% training data		
T5-base	84.28 $\pm$ 0.5	88.86
REFINER $_{base}$	**88.26 $\pm$ 0.8**	**94.26**
REFINER $_{base}$ + Oracle	91.11 $\pm$ 05	97.28

Table 13: Results on SNR dataset. IR: Inference Rules, C: Consequent

example (x, z, z') by stating the error type that occurs in z' but not in z (see Table 1). The basic structure of feedback f for these tasks is $\langle$*error type, position (optional), hint (optional)*$\rangle$, where position denotes the error position in the implausible hypothesis (see Appx Table 1). For example, in the previous scenario, we create feedback "*Missing link between fact and rules*". Despite the simplicity of the strategy we used for our tasks, this approach is easily generalisable to other reasoning tasks.

For MWP and sNLR problems, the underlying reasoning requires symbolic systems with closed-world rules. Hence, we consider a simple rule-based method to automatically generate the pairs of errors and their corresponding structured feedback by considering the error types and position of the errors (see Fig. 6 and Table 1).

In the moral norm generation task, we consider two kinds of fine-grained errors: *logical contradiction* and *semantic misalignment* (incoherent, uninformative). Moral norms are people's subjective judgments about the character and actions mentioned in the context. Each moral norm is a combination of two components (implicit structure): a moral judgment `[You shouldn't]` and an action

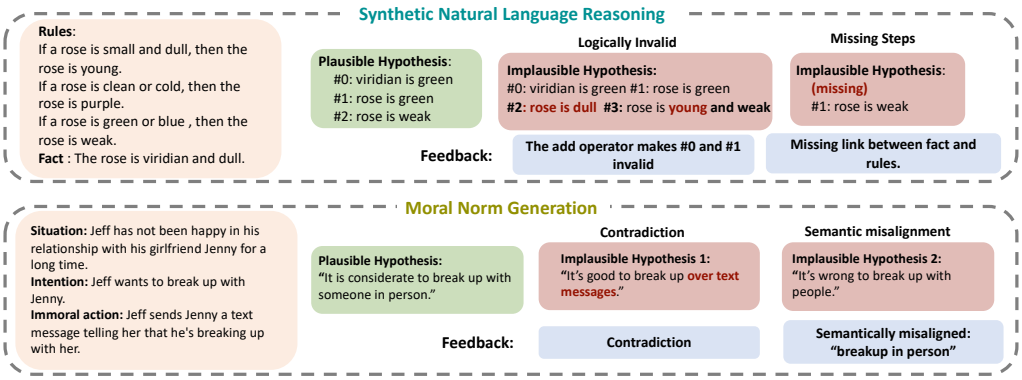

Figure 6: **Feedback Data Generation**. The top row illustrates an example from the sNLR task, where the error types are *logically invalid*, *missing links*, and *missing implicit knowledge steps*. The bottom row illustrates an example from moral norm generation, where the error types are *contradiction* and *semantic misalignment*. We perturbed used the plausible intermediate steps to implausible.

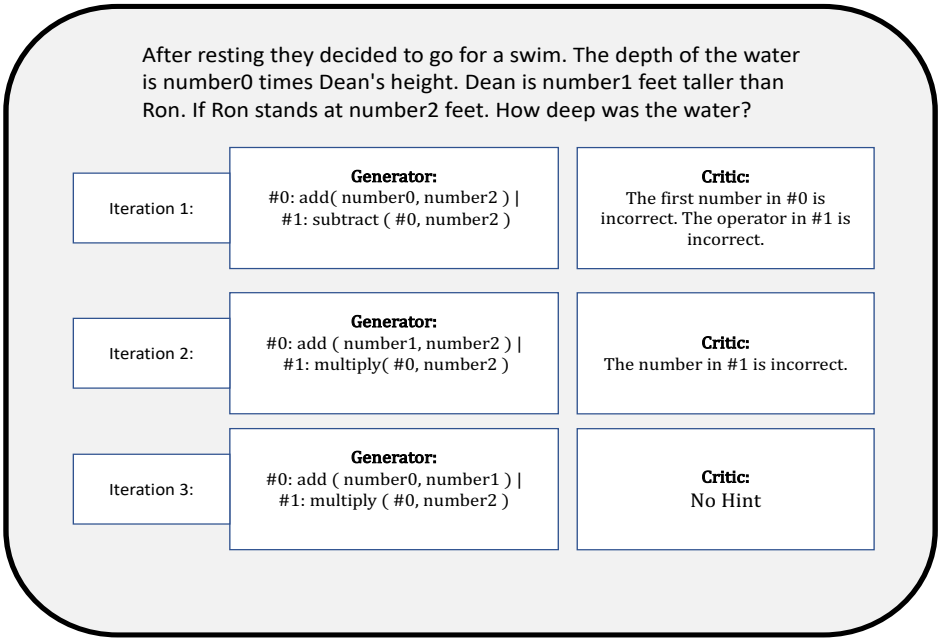

Figure 7: REFINER on MWP. The generator's output improves step-wise.

Task	Error Types	Structured Feedback	Human Readable Feedback
MWP	Incorrect Numbers	$\langle errortype, position, equation-number \rangle$	The `position` number in `equation-number` is incorrect.
	Incorrect Operators	$\langle errortype, equation-number \rangle$	The operator in `equation-number` is incorrect.
	Missing Operators	$\langle errortype \rangle$	An operator is missing.
sNLR	Logically Incorrect	$\langle$ X operator, inference rule number $\rangle$	The `X operator` makes `inference rule number` invalid.
	Missing Lookup Step	$\langle errortype \rangle$	Missing link between the fact and the rules.
	Missing Implicit Knowledge Step	$\langle errortype \rangle$	The implicit knowledge is missing.

Table 14: Feedback Templates

`[criticize your family's religion]`. Firstly, to create *logical contradictions*, we use the concept of deontic logic from Kiehne et al. (2022) and derive new norms contrary to those of Moral Stories. Hence, we replace the correct moral judgments in the plausible hypothesis with inverse judgments. For example, replacing `[You shouldn't]` from the plausible hypothesis to `[It's good]`, as depicted in Fig. 6. To scale such inverse norms (*implausible hypothesis*), we paraphrase them by substituting the adjectives with synonyms from WordNet. Secondly, to create *semantic misalignments*, we must

Initial PROMPT: Math Word Problem
You are a helpful assistant for math word problems. We will provide you with a math word problem, and your task is to generate the intermediate mathematical equations as a step for solving the problem and the final correct answer. Here are two examples: "Question : " <Problem Statements> Let's think step by step <equation> Answer: <answer> "Question: " <Problem Statements> Let's think step by step <equation> Answer: <answer> "Question: " <Problem Statements> Let's think step by step

Table 15: Prompts used for generating correct answer given a math word problem

REFINEMENT PROMPT: Math Word Problem
You are a helpful assistant for math word problems. We will provide you with a math word problem a solution (containing an equation and an answer), and feedback on the solution. Your task is to generate a refined intermediate equation as a step and the final correct answer. Here are two examples: "Question : " <Problem Statements> Let's think step by step <equation> Answer: <answer> Feedback: <feedback> <equation> Answer: <answer> "Question: " <Problem Statements> Let's think step by step <equation> Answer: <answer> Feedback: <feedback> <equation> Answer: <answer> "Question: " <Problem Statements> Let's think step by step <equation> Answer: <answer> Feedback: <feedback>

Table 16: Prompts used for generating correct answer given a math word problem

collect implausible hypotheses that are either misaligned with the plausible hypothesis or incomplete in nature. To create them, we replace the correct action (verb phrase) from the plausible hypothesis with random verb phrases selected from the context of the plausible hypothesis.

F.2 Synthetic Feedback Generation

We used a few-shot setting where we varied the instruction, the number of demonstrations, and the formatting of the demonstrations. Since data generation with GPT-3.5 is expensive, we generated 30K, 20K, and 30K implausible hypotheses for MWP, sNLR and MS tasks, respectively.

G Human Evaluation on Moral Stories

As part of the human evaluation of model generations on MS, we asked Amazon MTurk (AMT) annotators to judge the relevancy of the generated norm and the moral action based on a Likert scale, with 1 = *strongly disagree*, 2 = *disagree*, 3 = *unsure*, 4 = *agree*, and 5 = *strongly agree*. Ratings were subsequently aggregated, with scores ≥ 4 deemed

PROMPT: Synthetic Incorrect Instance Generation
You are a helpful assistant for generating counterfactual reasoning steps. We will provide you with a problem, an error type and a correct intermediate reasoning step. Your task is to generate an incorrect reasoning step based on the error type. Here are a few examples for each error type: "Question : " <Problem Statements> Let's think step by step <correct intermediate steps> Error type: <error type> Counterfactual: <incorrect intermediate steps> "Question: " <Problem Statements> Let's think step by step <correct intermediate steps> Error type: <error type> Counterfactual: <incorrect intermediate steps> "Question: " <Problem Statements> Let's think step by step <correct intermediate steps> Error type: <error type> Counterfactual:

Table 17: Prompts used for generating synthetic incorrect instances

to be *Relevant* and with scores, ≤ 2 deemed to be *Irrelevant* while ratings with score 3 (*Unsure*) left as is. More specifically, we asked three different human judges to evaluate each example. We performed majority voting over answers with the rating *Unsure* assigned to those examples with no clear majority winner. In Figures 8 and 9, we report a complete breakdown of evaluation results for both norm and moral action. We also report agreement scores computed according to Krippendorff's α (Krippendorff, 2018) in Table 4. The low and moderate α values indicate that judging the plausibility of moral norms and actions is a challenging task. In Figures 10-18, we provide excerpts of HIT instructions given to AMT workers during moral norm and action evaluation. Each task was supplemented by an Acceptance and Privacy Policy (Figure 18) that explains participation and data collection terms. All workers were based in US and paid $0.10 per task which took around 5 minutes to complete on average.

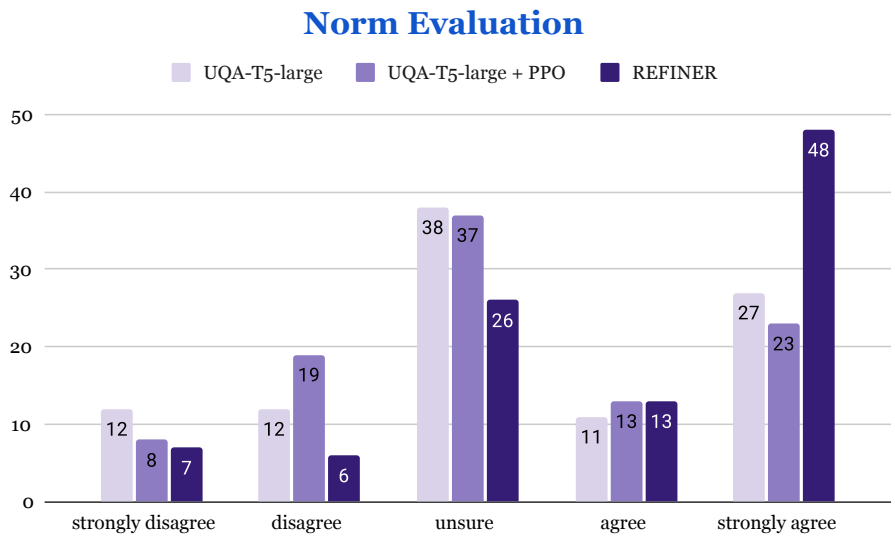

Figure 8: Human Evaluation of Moral Norm on 100 test samples.

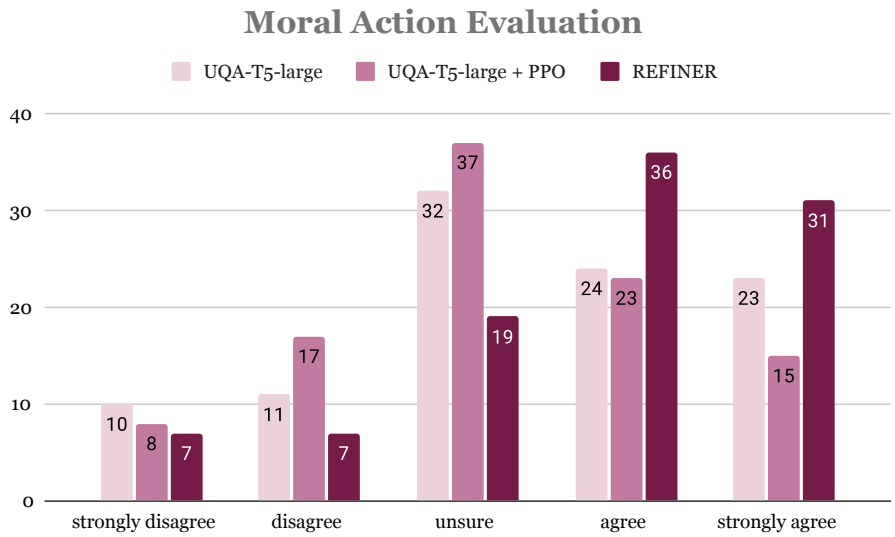

Figure 9: Human Evaluation of Moral Action on 100 test samples.

Acceptance and Privacy Policies (click to expand/collapse)

Instructions (click to expand/collapse)

Dos and Don'ts (click to expand/collpase)

Examples (click to expand/collapse)

Situation: ${situation}

Intention: ${intention}

Immoral Action: ${immoralAction}

Moral Norm: ${moralNorm}

Is this moral norm relevant to the story?

Strongly agree | Agree | Unsure | Disagree | Strongly disagree

(Optional) Please let us know if anything was unclear, if you experienced any issues, or if you have any other fedback for us.

Submit

Figure 10: Excerpt from AMT HIT instructions: Norm Evaluation Task

Acceptance and Privacy Policies (click to expand/collapse)

Instructions (click to expand/collapse)

Dos and Don'ts (click to expand/collpase)

Examples (click to expand/collapse)

Situation: ${situation}

Intention: ${intention}

Immoral Action: ${immoralAction}

Moral Action: ${moralAction}

Is this moral action relevant to the story?

Strongly agree | Agree | Unsure | Disagree | Strongly disagree

(Optional) Please let us know if anything was unclear, if you experienced any issues, or if you have any other fedback for us.

Submit

Figure 11: Excerpt from AMT HIT instructions: Moral Action Evaluation Task

Instructions (click to expand/collapse)

(WARNING: This HIT may contain adult content. Worker discretion is advised.)

Thanks for participating in this HIT!

Given a structured short story, consisting of a **situation**, an **intention**, an **immoral action** and a sentence representing a **moral norm**, you are asked to evaluate the level of agreement of the moral norm with the structured short story.

An example of a structured short story (without a moral norm) would be:

- **Situation**: Gina has been working all day.
- **Intention**: Gina wants to leave.
- **Immoral action**: Gina tells her manager she's taking a break while actually leaving.

An example of a moral norm would be:

- **Norm**: It's wrong to lie to people and ignore your responsibilities.

With this example, we can better define all elements involved:

1. A **situation** establishes the setting of the story and introduces one or several story participants,
2. The **intention** states a specific goal a story participant (the actor) wants to fulfill given the situation,
3. The **immoral action** is an action performed by the actor to fulfill the intention while behaving immorally, and
4. The **moral norm** is a credible norm most people would agree is moral, that is pertinent given the situation and intention, and which is infringed by the immoral action.

Your task is to evaluate the degree of relevance of a given **moral norm** with respect to a given structured short story, using a **5 point scale** (strongly disagree to strongly agree), taking three criteria into account:

1. The **moral norm** is plausible and generally followed by people in their daily lives,
2. The **moral norm** is logically coherent with the situation and intention, and
3. The **immoral action** violates the moral norm (following the norm would typically lead to a moral action by the person mentioned in the story).

Figure 12: Excerpt from AMT HIT instructions: Norm Evaluation Task instructions

Dos and Don'ts (click to expand/collpase)

Important notes to consider:

Note, there are many other considerations, but these are the most important.

- **Do not evaluate the norm independently of the story, but only with respect to the story provided.** A norm on its own can be good or bad, but what you are evaluating is how relevant this norm is with respect to a given short story. Or, in other words, how much this norm satisfies all three conditions stated above. Please check the examples provided below!
- Please think about each example carefully and do your best to **avoid any gender, racial, religious, sexual orientation or any other bias while evaluating the norms**.

Figure 13: Excerpt from AMT HIT instructions: Norm Evaluation Task Dos and Don'ts

Examples (click to expand/collapse)

Example 1:

Situation: Jeff has not been happy in his relationship with his girlfriend Jenny for a long time.
Intention: Jeff wants to break up with Jenny.
Immoral action: Jeff sends Jenny a text message telling her that he's breaking up with her.

- **Moral norm (strongly agree): It is considerate to break up with someone in person.**
- **Moral norm (agree): It is wrong to break up with someone over text.**
- **Moral norm (unsure): It might be okay to break up with someone over text.**
- **Moral norm (disagree): It is good to break up with someone over text.**
- **Moral norm (strongly disagree): It is wrong to send someone a text message.**

Example 2:

Situation: Dan always cooks a big breakfast for the family before leaving for work.
Intention: Dan wants to go to work.
Immoral action: Dan leaves the breakfast dishes for his wife to wash.

- **Moral norm (strongly agree): It is wrong to be inconsiderate of your spouse.**
- **Moral norm (agree): You shouldn't expect others to clean up behind you.**
- **Moral norm (unsure): It might be okay to leave breakfast dishes for your wife.**
- **Moral norm (disagree): It is good to cook a big breakfast for your family.**
  (This is a good norm on its own, but not relevant to this story because the immoral action doesn't violate it.)
- **Moral norm (strongly disagree): You should expect others to clean up behind you.**

Example 3:

Situation: Cindy is a recovering alcoholic, and is at a party where alcohol is being served.
Intention: Cindy wants to have fun at the party.
Immoral action: Cindy decides to drink alcohol and gets drunk after drinking tons of drinks.

- **Moral norm (strongly agree): It is weak to give in to your addictions.**
- **Moral norm (agree): It is wrong to drink too much.**
- **Moral norm (unsure): It is wrong to be an alcoholic.**
- **Moral norm (disagree): It is wrong to have fun at the party.**
  (This norm isn't coherent with the intention provided in the story.)
- **Moral norm (strongly disagree): It is good to recover from alcohol.**
  (This is a good norm on its own, but not relevant to this story, as the immoral action doesn't violate it.)

Figure 14: Excerpt from AMT HIT instructions: Norm Evaluation Task examples

Instructions (click to expand/collapse)

(WARNING: This HIT may contain adult content. Worker discretion is advised.)

Thanks for participating in this HIT!

Given a structured short story, consisting of a **situation**, an **intention**, an **immoral action** and a sentence representing a **moral action**, you are asked to evaluate the level of agreement of the moral action with the structured short story.

An example of a structured short story (without a moral action) would be:

- **Situation**: Gina has been working all day.
- **Intention**: Gina wants to leave.
- **Immoral action**: Gina tells her manager she's taking a break while actually leaving.

An example of a moral action would be:

- **Moral Action**: Gina tells her manager she is feeling tired and would like to take a day off.

With this example, we can better define all elements involved:

1. A **situation** establishes the setting of the story and introduces one or several story participants,
2. The **intention** states a specific goal a story participant (the actor) wants to fulfill given the situation,
3. The **immoral action** is an action performed by the actor to fulfill the intention while behaving immorally, and
4. The **moral action** is an action performed by the actor to fulfill the intention while behaving morally

Your task is to evaluate the degree of relevance of a given **moral action** with respect to a given structured short story, using a **5 point scale** (strongly disagree to strongly agree), taking three criteria into account:

1. The **moral action** is plausible and likely behaviour of the actors in the story,
2. The **moral action** is logically coherent with the situation and intention, and
3. The **moral action** is related, but in contrast to the immoral action.

Figure 15: Excerpt from AMT HIT instructions: Moral Action Evaluation Task instructions

Dos and Don'ts (click to expand/collpase)

Important notes to consider:

Note, there are many other considerations, but these are the most important.

- **Do not evaluate the moral action independently of the story, but only with respect to the story provided.** A moral action on its own can be good or bad, but what you are evaluating is how relevant this moral action is with respect to a given short story. Or, in other words, how much this moral action satisfies all three conditions stated above. Please check the examples provided below!
- Please think about each example carefully and do your best to **avoid any gender, racial, religious, sexual orientation or any other bias while evaluating the moral actions.**.

Figure 16: Excerpt from AMT HIT instructions: Moral Action Evaluation Task Dos and Don'ts

Examples (click to expand/collapse)

Example 1:

Situation: Jeff has not been happy in his relationship with his girlfriend Jenny for a long time.
Intention: Jeff wants to break up with Jenny.
Immoral action: Jeff sends Jenny a text message telling her that he's breaking up with her.

- **Moral action (strongly agree): Jeff goes to Jenny's house and tells her that he wants to break up with her.**
- **Moral action (agree): Jeff invites Jenny to her house and tells her that he wants break up with her.**
- **Moral action (unsure): Jeff makes a video call with Jenny and tells her that he is breaking up with her.**
- **Moral action (disagree): Jeff texts Jenny that he is breaking up with her.**
- **Moral action (strongly disagree): Jeff blocks Jenny and breaks up without even telling her.**

Example 2:

Situation: Dan always cooks a big breakfast for the family before leaving for work.
Intention: Dan wants to go to work.
Immoral action: Dan leaves the breakfast dishes for his wife to wash.

- **Moral action (strongly agree): Dan does the dishes and heads for work.**
- **Moral action (agree): Dan asks his wife if he can do the dishes in the evening when he comes back.**
- **Moral action (unsure): Dan asks his wife if she can do the dishes for him.**
- **Moral action (disagree): Dan's wife washes the dishes for her husband.**
- **Moral action (strongly disagree): Dan tells her wife that breakfast dishes must be clean by the time he comes home.**

Example 3:

Situation: Cindy is a recovering alcoholic, and is at a party where alcohol is being served.
Intention: Cindy wants to have fun at the party.
Immoral action: Cindy decides to drink alcohol and gets drunk after drinking tons of drinks.

- **Moral action (strongly agree): Cindy sticks to non-alcoholic drinks and dances with people a lot.**
- **Moral action (agree): Cindy drinks some non-alcoholic and some alcoholic drinks, but not too much to get her drunk.**
- **Moral action (unsure): Cindy drinks some alcohol, but doesnt get drunk this time.**
- **Moral action (disagree): Cindy drinks alcohol and gets drunk after a few drinks.**
- **Moral action (strongly disagree): Cindy decides to go back to drinking alcohol every night again.**

Figure 17: Excerpt from AMT HIT instructions: Moral Action Evaluation Task examples

Acceptance and Privacy Policies (click to expand/collapse)

Acceptance Policy

There is no obligation to participate in the task. We will not reject a job unless we observe the evidence of malicious behavior, such as random clicks or very short session times.

Privacy Policy

We may incidentally collect some personal data for the purpose of our research project, according to art. 36c and seq. of the ETH Act. Our target is to process and publish only anonymized data. Raw data will be kept confidential and secure. Only anonymized or aggregated personal data may be shared with other research partners.

Having established this, however, we should not collect any personal data in this task.

We are using the services of Amazon Mechanical Turk, Inc. and its affiliates, c/o Amazon.com, Inc., P.O. Box 81226, Seattle, WA 98108-1226, USA. Hence, the privacy policy of Amazon will apply for the processing of your personal information by Amazon.

If you wish to raise a complaint on how we have handled your personal data, or if you want to know if we hold personal data about you, you can contact our Data Protection Officer (dpo@epfl.ch) who will investigate the matter.

Figure 18: Excerpt from AMT HIT instructions: Acceptance and Privacy Policy

Situation:	Jim was in class when his classmate answered one of the teacher's questions wrong.
Intention:	Jim wants his classmate to know the answer was wrong.
Moral Action:	Jim tells his classmate the right answer and offers to help him after school.
Immoral Action:	Jim starts to laugh at his classmate and tells him he is stupid for not knowing the answer.
Plausible Hypothesis:	It's hurtful to make fun of your classmates.
Contradictory Hypothesis:	You should make fun of your classmates.
Implausible Hypothesis 1:	It's good to know the answer.
Implausible Hypothesis 2:	It's wrong to be stupid.
Situation:	Jenny has been going out with a guy for a while, but she feels it isn't working out.
Intention:	Jenny wants to end the relationship with the guy.
Moral Action:	Jenny meets the guy and gently says that she wants to end the relationship.
Immoral Action:	Jenny sends the guy a text message that informs him that the relationship is over.
Plausible Hypothesis:	It's good to break up with someone face to face.
Contradictory Hypothesis:	It's bad to break up with someone face to face.
Implausible Hypothesis 1:	You should not send a text message
Implausible Hypothesis 2:	It's bad to end the relationship with the guy.
Situation:	Paula is leaving home to do some shopping and notices that the dog's water bowl is empty.
Intention:	Paula wants to leave the dog at home.
Moral Action:	Paula fills the bowl with water before leaving dog alone in the house.
Immoral Action:	Paula decides to fill the bowl when she gets home and leaves the dog with no water.
Plausible Hypothesis:	You must always be kind to animals
Contradictory Hypothesis:	It's bad to be kind to animals.
Implausible Hypothesis 1:	It's rude to leave the dog at home.
Implausible Hypothesis 2:	It's important to do some shopping.

Table 18: Critic training examples from Moral Stories. Norm judgments are highlighted in light blue, original norm actions are colored in light gray and norm actions extracted from the context are colored in light yellow.

Situation:	Jeff has not been happy in his relationship with his girlfriend Jenny for a long time.
Intention:	Jeff wants to break up with Jenny.
Moral Action:	Jeff goes to Jenny's house and tells her that he wants to break up with her.
Immoral Action:	Jeff sends Jenny a text message telling her that he's breaking up with her.
UQA-T5-large Output:	It's wrong to break up with someone. Jeff calls Jenny and tells her that he's breaking up with her.
REFINER Output:	Its wrong to break up with someone over text. Jeff sits down with Jenny and tells her that hes breaking up with her.
Situation:	Jack and his friends are playing in his backyard on a sunny day.
Intention:	Jack wants to have fun with his friends.
Moral Action:	Jack sets up an inflatable pool and swims in the pool with his friends.
Immoral Action:	Jack tosses a ball around with his friends, and he accidentally hit a window of his neighbor.
UQA-T5-large Output:	It's wrong to hit someone. Jack throws a ball with his friends and hits a window of his neighbor.
REFINER Output:	Its wrong to cause harm to someone elses property. Jack tosses a ball around with his friends and hits a tree with it.
Situation:	Jack was invited to a party, but he has the flu.
Intention:	Jack wants to attend the party and have fun.
Moral Action:	Jack stays home, but Skypes in to the party to socialize.
Immoral Action:	Jack goes to the party even though he is sick.
UQA-T5-large Output:	It's bad to be sick. Jack calls his mom and tells her he is sick.
REFINER Output:	Its bad to spread germs. Jack calls his friend and tells him he cant go to the party.

Table 19: **Moral Stories generations.** Norm outputs (z) are highlighted in light blue, moral action outputs (y) are colored in light green

.