Parallel Scaling with Entropic Reasoners

Authors. Devan Shah, Owen Yang, Daniel Yang

Link to GitHub Repo. https://github.com/dshah02/EntropicReasoners

Abstract

When asked the same question many times, a large language model will often respond similarly each time, not taking full advantage of the multiple attempts. However, in some settings, such as code generation with unit tests or proof generation in a formal verifier, the model only requires a single correct answer out of any amount of tries to have adequately helped the user. Often, it is acceptable to run a language model multiple times if it increases the likelihood that the most common answer is correct. We investigate a novel application of Mutual Information Skill Learning to large language models, allowing users to ensure language model response diversely by using different conditioning vectors $z \sim \mathcal{Z}$. This allows for diverse, useful LLM responses without the need for domain-knowledge or manually prompt engineering. We test our technique on GSM8K [2] and a variety of large language models, finding small improvements across some models on pass@k and consensus@k tasks. We also provide insight into improvements needed for better performance.

1 Introduction

1.1 Context

Recent advancements in large language models (LLMs) have demonstrated their capabilities across multiple domains, including on tasks involving reasoning steps [5]. However, these models often exhibit limited output diversity across multiple trials—that is, they will produce similar responses when prompted multiple times on the same task [14]. This output homogeneity presents a significant roadblock in the ability of models to solve complex reasoning tasks, where exploring multiple solution paths greatly increases the chance of arriving at the correct answer and reduces the likelihood that a single hallucination ruins the reasoning chain.

Multi-Attempt Metrics. In particular, the evaluation of model performance on complex reasoning or generation tasks has increasingly adopted metrics that take the best out of multiple attempts, like pass@k and cons@k. These metrics help account for the fact that for many verifiable problems, such as mathematical proofs or code generation, obtaining a single correct solution across multiple attempts is sufficient, as it can be verified with automated programs such as a compiler. For these tasks, the ability to generate diverse and varied solutions increases the likelihood of exploring a correct reasoning path at least once, even if all other attempts fail.

Several modern state-of-the-art problem-solving AI systems use this approach. For instance, AlphaGeometry [18] leverages a pass@512 beam width to solve Olympiad-level geometry problems, and AlphaCode [10] is effectively pass@timeout, generating hundreds of thousands of candidate solutions and whittling them do the best ones. In common chat models, GPT-4o [7] and OpenAI o1 [8] use pass@k metrics to evaluate both image generation as well as math and coding abilities.

1.2 Challenges

Current Limitations. Currently, the best way to encourage greater diversity in model responses across several trials is to employ different prompts for each trial [16]. For

instance, for a challenging math problem, we may prompt the model to use algebra in one trial, geometry in another, trigonometry in a third, etc. However, for more complicated tasks, this approach requires advanced domain knowledge to determine which techniques will be helpful and lead to qualitatively different responses. For example, we may not know that algebra is irrelevant to this problem, or that geometry and trigonometry lead to the same solution path.

Reasoning training algorithms such as Group Relative Policy Optimization (GRPO) [15] have become increasingly popular for improving model performance. These algorithms only require datasets of questions with easily verifiable answers to train high-performing reasoning models [15]. Our technique is designed to complement such reasoning training approaches, requiring only question-solution pairs with verifiable solutions. This limits us for training the model on several different techniques to try each problem, as we do not have access to the problem solving strategies for each problem—only the correct answer. While current reasoning training incentivizes language models to learn problem-solving strategies, it lacks mechanisms for encouraging models to develop multiple distinct approaches, and this so far has been an underexplored area in current research.

Prior Work and Our Contributions. Prior methods [11] for improving pass@k performance typically generate far more than k solutions and then filter down to the best k candidates. This line of work is largely orthogonal to our research direction, as we do not subsample model responses but instead aim to maximize the utility of each generated solution through increased diversity.

In this work, we introduce novel reward techniques that specifically encourage language models to learn diverse reasoning strategies during GRPO reasoning training. We evaluate our approach on three leading open-weight models: Qwen 2.5-7B, Llama 3.1-8B, and R1-Distilled-Qwen2.5-Math-1.5B. We fully open-source our codebase at `dshah02/EntropicReasoners` ↗. See Figure 6 for an high-level sample comparison.

2 Problem Background

2.1 Mutual Information Skill Learning

Self-Supervised Skill Learning. Mutual Information Skill Learning [20] is a technique within the broader reinforcement learning research direction of self-supervised skill learning. For environments with large amounts of unlabeled data and no reward system, self-supervised skill learning provides a technique for models to improve in performance and develop an understanding of the system around them with minimal human intervention and without the limitation of human priors. This allows for less reliance on annotated training data and the prospect of super-human performance.

To formalize this approach, let us consider an agent in an environment with states $s \in \mathcal{S}$, with initial state s_{start} and terminal state s_{end} and actions $a \in \mathcal{A}$ and environment transition distribution $p(s'|s,a)$. In state s, the actor samples their next action from the probability distribution $\pi(a|s)$, where π represents the agent's policy. Starting from s_{start}, we define the sequence of states and actions $(s_{\text{start}}, a_1, s_2, a_2, \ldots, s_{\text{end}})$ as the model's trajectory. Given the policy π and the environment transition function, we can determine the probability of each trajectory $\tau = (s_{start}, a_1, \ldots, s_{end})$ (under the policy π) as $\mathbb{P}_{\pi,p}[\tau] = \prod_{i=1}^{T-1} \pi(a_i|s_i)p(s_{i+1}|s_i,a_i)$. We will refer to this distribution over trajectories as $T(\pi)$.

Core MISL Objective. Mutual information skill learning proposes that, for some conditioning set $\mathcal{Z}$, we learn a policy conditioned on z $\pi(\cdot|s,z)$ for $z \in \mathcal{Z}$ such that

$$\pi = \arg\max_{\pi'} \left(\mathcal{I}_{z\sim\mathcal{Z},\tau\sim T(\pi(\cdot|z))}(\tau; z) \right) \tag{1}$$

where $\mathcal{I}$ is the mutual information between the distributions, defined as $\mathcal{I}(X,Y) := -\mathcal{H}(X|Y) - \mathcal{H}(X)$, with $\mathcal{H}(\cdot) := \mathbb{E}[-\log \mathbb{P}[X]]$ referring to Shannon Entropy over discrete distributions.

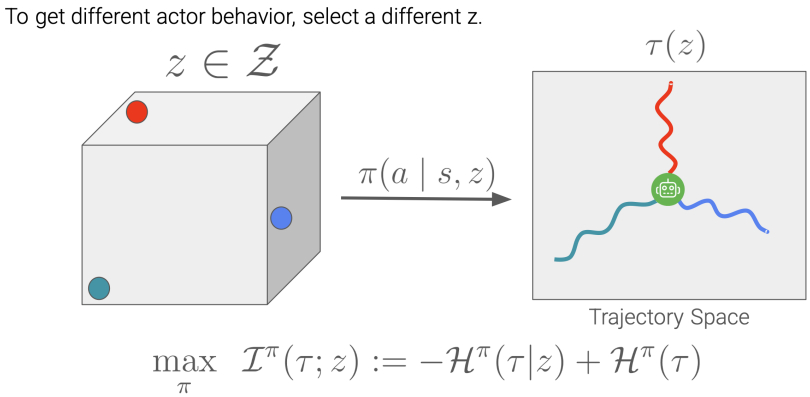

Figure 1: The trajectories are embedded into some trajectory space, which hopefully are very different for z that are far apart in $\mathcal{Z}$; in particular if $\mathcal{Z}$ is discrete then all strategies hopefully generate very different trajectory distributions.

Intuition and Interpretation. The fundamental goal of mutual information skill learning is to allow the model to learn a distinct "skill" for each $z \in \mathcal{Z}$, where the "skills" refer to policies that explore different parts of the state space. To better understand Equation 1, we can decompose the mutual information term into two components that need to be optimized. Our objective is to maximize $-\mathcal{H}(\tau|z) + \mathcal{H}(\tau)$.

To maximize the term $\mathcal{H}(\tau)$, we require that our model learn policies that explore a large amount of state space and have good coverage over the possible trajectories. To minimize the term $\mathcal{H}(\tau|z)$, we hope that with the knowledge of z, our conditioning term, there is minimal uncertainty in the trajectory the model will take. In other words, z controls the model trajectory and thus we can encourage the model to explore diverse states by running the policy on distinct z. In a sense, the diversity and randomness of our policy can entirely be attributed to the randomness of choosing $z \sim \mathcal{Z}$.

Thus, mutual information skill learning provides a technique to encourage model diversity and allow us to ensure diverse policies by sampling distinct z.

2.2 Large Language Models as RL Agents

Large Language Models learn a mapping ϕ and matrix W^o that allows the models to determine a distribution over the next token based on all the previous tokens in context, where tokens and a tokenizer provide a certain chunking of the input text. To predict such a distribution, the model learns ϕ : text $\to \mathbb{R}^h$, for some large h, and then a matrix $W^o \in \mathbb{R}^{h \times |V|}$, where V is the set of tokens. Thus, by computing $s_{max}(W^o \phi(\text{input text}))$ we have a distribution over the $|V|$, where s_{max} is the standard softmax function $s_{max}(\vec{x}) = \left(\frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}\right)_i$.

How do LLMs Generate Responses? To generate text after a prompt, we can then iteratively sample from the distribution of next tokens produced by the language model to produce coherent text, like in Figure 2.

Language models are trained in a self-supervised manner from a large corpus of text data with a goal of minimizing the cross entropy loss between its predicted distribution over the next token and the actual next token.

Language Models as MDPs. However, note that language models can also be viewed as learning a policy over a certain Markov Decision Process. Each state is the current tokens the model has seen, and the model learns a policy π that associates each state with a distribution over next tokens, with the token chosen representing the action. The transition

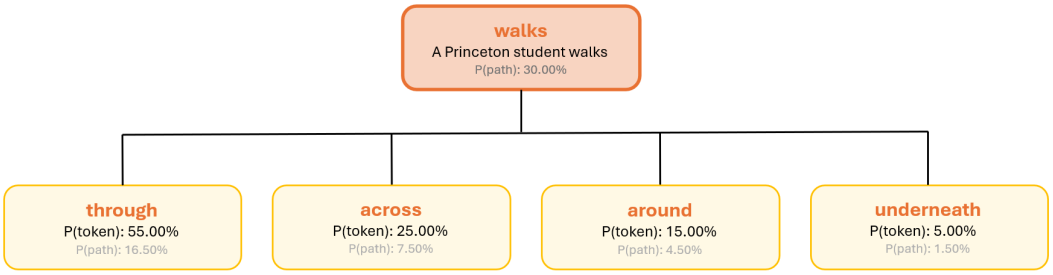

Figure 2: Potential sampling of a language model completing the next word in the sentence: "A Princeton student walks".

function simply maps each state-action pair to the new state created by combining the prior text with the new token chosen. To improve model performance and produce models with more preferred outputs, language models can then be ascribed a reward at the end of each text completion and this reward can be used to update the policy just as with traditional reinforcement learning to encourage better responses.

A common method for improving language model performance on problem solving tasks is to give a reward corresponding to whether a model correctly answered a question. This reward signal, together with the GRPO algorithm, have led to the development of strong language models and training techniques that simply require a list of questions and answers that can be easily numerically verified, such as with math problems.

3 Methods

We introduce two different rewards based on the principle of maximizing mutual information. For both, we consider the conditioning set to be $\mathcal{Z} = \{1, 2, \ldots, N\}$ for some integer N. Both of these methods will be used in addition to GRPO training to create math problem solving agents. Additionally, to condition the language model on $z \in \mathcal{Z}$, we append the text "`Strategy` $\{z\}$" before each prompt.

3.1 Token Mutual Information

To stay close with prior work, we consider the language model as an reinforcement learning agent over text states as described in section 1.3.2. To encourage mutual information skill learning, we add an additional reward term during GRPO training that corresponds to the mutual information of each output trajectory, which we call the **token mutual information reward**.

If we permit multiple starting states s_{start}, which for our LLM correspond to prompts $x \in \mathcal{X}$, then Equation 1 is equivalent to maximizing:

$$\mathbb{E}_{\tau \sim T(\pi), z \sim \mathcal{Z}, x \in \mathcal{X}}[\ln \mathbb{P}[\tau|x, z] - \ln \mathbb{P}[\tau|x]]$$

We aim to construct a mutual information reward that also corresponds to maximizing this objective. For each input prompt x, we randomly select $z \sim \mathcal{Z}$, and the model generates C trajectories $\tau_1, \tau_2, \ldots \tau_C$. We then compute the reward $r(x) = \sum_{i=1}^{C} (\ln \mathbb{P}[\tau_i|x, z] - \ln \mathbb{E}[\tau_i|x])$ based on those trajectories. The token mutual information reward is then added to the traditional GRPO reward with weight parameter α_1, where the traditional GRPO reward includes a binary verifiable reward and a KL-divergence penalty to ensure generation does not veer far from the base model.

3.2 Semantic Mutual Information

The above reward prioritizes different trajectories over token space. However, often times, differences in token-space, such as choosing different formatting or permuting words, do not correspond to different high-level reasoning strategies.

Thus, we crafted a second mutual information reward that uses a projection mapping ψ to embed each output. This mapping ψ serves as a compression of the trajectory and thus considering the mutual information between $\psi(T(\pi))$ and $\mathcal{Z}$, for a well-suited projection, should retain the semantic meaning of the token completion while being less sensitive to the specific syntactic way the message is expressed. In practice, for the embedding function ψ, we will choose a pretrained large language model for general text embedding. Because this reward takes into account semantic meaning, we call it the **semantic mutual information reward**.

As we no longer can compute $\mathbb{P}[\psi(\tau)|x]$, we now need a different technique to estimate the mutual information. We make the simplifying assumption that ψ is a continuous map to some semantic embedding space and use the Kraskov-Stögbauer-Grassberger (KSG) k-NN estimator for mutual information of continuous distributions to approximate the semantic mutual information [9]. For a brief explanation of this estimator, see Appendix A.

Similar to the standard MI reward, we use the weight parameter α_2 to add the semantic MI reward to the entire reward function, which includes the traditional GRPO reward and the token MI reward.

4 Experiments

Base Models Used. For our control models, we train each of the three base models in Table 1 with GRPO using 2000 training examples from GSM8K [2], limiting the maximum token sequence length to 1024. Rather than full parameter updates, we used the Unsloth library to train LoRAs of approximately 80M parameters for each model [3]. We additionally use Unsloth for the GRPO training, with our codebase extending their implementation [19].

Table 1: LLMs used in this paper

Model Name	Params	Open-source	MMLU	MATH	GSM8K
Llama 3.1	8B	✓	73.0	73.8	96.8[1]
Qwen2.5	7B	✓	74.2	49.8	85.4[2]
R1-Qwen2.5[3]	1.5B	✓	-	83.9	-

[1] 8-shot result [2] 4-shot result

[3] Deepseek fine-tuned the Qwen2.5-Math-1.5B base model with 800K samples of reasoning generated by Deepseek-R1.

Our Investigations. For our experimental models, we then added the token and semantic mutual information reward terms to the GRPO training reward of correctness, as explained in subsection 3.1 and subsection 3.2, which incentivize each model response to be different when given a different strategy number. For the semantic mutual information reward, we use the general text embedding model `multilingual-e5-large-instruct`, along with the NPEET library's implementation of the KSG estimator for mutual information [17].

We reran the training method with the modified reward functions, ablating the hyperparameters α_1 and α_2 as well as N, which control the reward function weightings and the number of strategies. When $\alpha_2 = 0$ we tested the settings $N = 5, 10, 20$. with all models. Otherwise, for the KSG estimator, we require at least $k + 1$ samples per strategy. Here we choose to generate 6 samples and so we must generate $6N$ completions total per estimate of semantic mutual information, with more completions allowing an estimate with less variance. As this presents a significant bottleneck in compute time, we only tested nonzero

values of α_2 for $N = 5$. In these cases we generated 6 completions per strategy, for a total of 30 completions per estimate of semantic mutual information.

We then measured the performance of the control and experimental models on 100 test prompts from the GSM8K dataset.

5 Results

5.1 Token Mutual Information

Figure 3 shows that with $z = 5$, $\alpha_1 = 5$, and $\alpha_2 = 0$, our mutual information framework can lead to substantial performance improvements on consensus at 5. Interestingly, we also notice an unexpected increase in the pass@1 performance, which we did not expect, which can be a subject of further investigation. Plurality did not increase significantly under the mutual information training either, indicating that the most common answer is not significantly better compared to the baseline. However, the consensus increase suggests that models have now become more confident in the correct answer. Our results fall far from the Table 1 results from Llama 3.1 and Qwen2.5. This may be due to different correctness measures: rather than 8 or 4-shot, we use zero-shot prompting; we also have answer extraction code that may be faulty and limit model output tokens to length 1024, which cuts off several model outputs before they finish solving the problem.

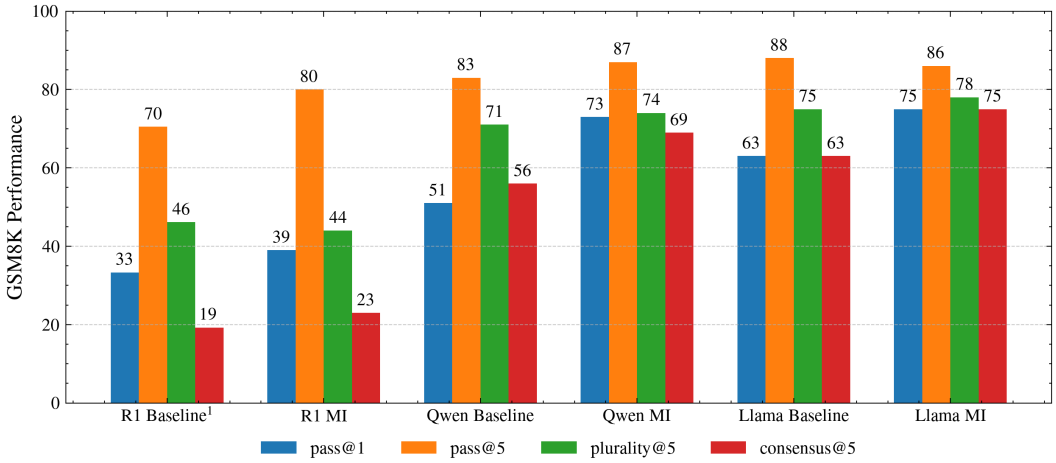

Figure 3: Model performance (percent correct), trained with $z = 5$.

[1] Evaluated on 78/100 problems due to reaching compute limits, results extrapolated for fair comparison.

In Figure 4, we fix the base model at Qwen2.5 and instead vary $\mathcal{Z}$. We measure pass@$|\mathcal{Z}|$, but note the largest improvement with $z = 5$. We conjecture that our hyperparameter search was most effective for $z = 5$, and additionally that for larger $\mathcal{Z}$, there are simply not enough diverse strategies to learn on the GSM8K questions, and this leads to training instability.

Through the course of our experiments, we have noticed that there is substantial variability in the GRPO training process. Each baseline required 5 hours of compute on an Nvidia H100 GPU and each experimental model required around 11 hours. Unfortunately, this meant we were unable to train enough models to properly account for this variability. Over the course of this project, we trained over 60 experimental models to test different techniques.

5.2 Token and Semantic Mutual Information

When attempting to train models with the parameter regime $\alpha_1 = 0$ and a nonzero value of α_2, all three models interestingly are unable to learn to maximize the semantic mutual

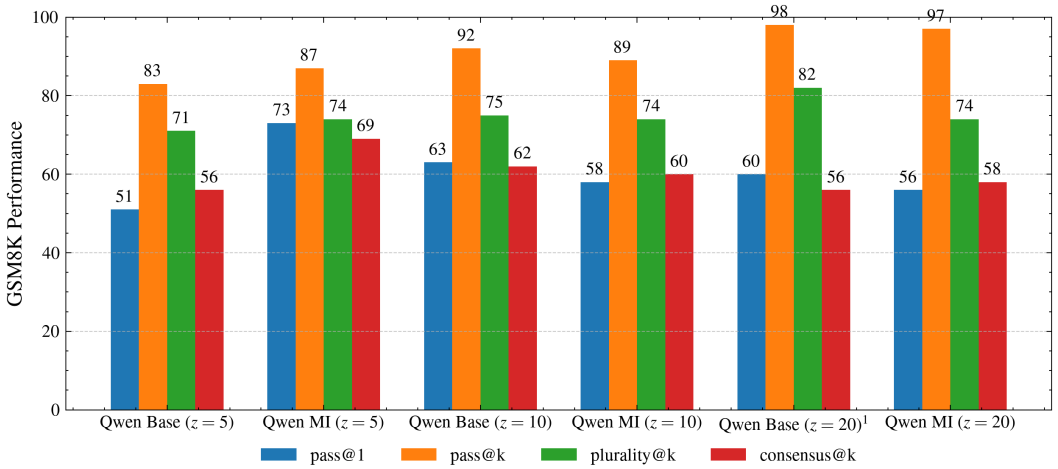

Figure 4: Model performance (percent correct), trained with varying z.

[1] Evaluated on 50 problems

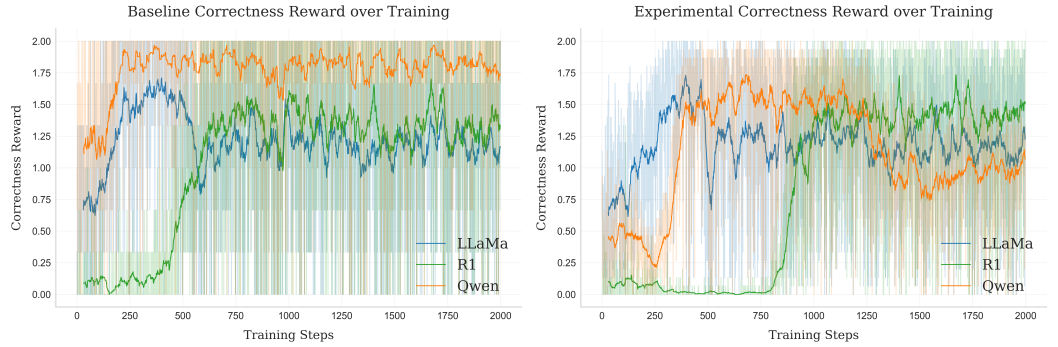

Figure 5: Correctness reward in training for parameter regime $(\alpha_1, \alpha_2) = (5, 5)$ and 2000 training steps. The reward is scaled from 0 to 2, corresponding to how many times the model is correct over 2 completions.

information reward; the semantic reward term oscillates around 0 for the entire training process. We hypothesize that this occurs because the reward signal to output ratio is simply too low. If it takes 30 completions to generate one reward signal for semantic mutual information, the sparsity of this signal may result in the model being unable to determine how to maximize it, especially when weighing it in relative importance compared to the much denser signal of the correctness reward.

Table 2 contains a summary of our (α_1, α_2) hyperparameter ablations. It is very interesting to note that for each fixed value of α_1, increasing α_2 always results in a higher pass@k evaluation for every model tested. Extrapolating this trend, we likely have not found the optimal parameter regime (α_1, α_2) that would result in the best-performing model for any of the three base modelsl. To test this, we would need to further raise the value of α_2 and test values of α_1 in between those listed in the table, where it seems the peak is attained. The Llama and R1 models that performed the best on pass@k were trained on the parameter regimes (α_1, α_2) equal to $(2, 2)$ and $(1, 3)$ respectively.

On the other hand, when adding in the semantic mutual information reward term, Qwen's performance degrades significantly. Accordingly, the best pass@k result for Qwen we trained had parameter regime $(\alpha_1, \alpha_2) = (5, 0)$, i.e. the semantic reward term was not used at all. According to the training graphs in Figure 5 and Figure 7, for which $\alpha_2 = 5$, Qwen suffered a sudden dropoff in all three rewards around step 1250 from which it failed to recover. When inspecting the sample outputs from this model, several strategies were

outputting incoherent text or repeating the same words, and the model appears stuck in this local minimum. This is despite a KL-diverge term in GRPO aiming to prevent such divergence. This drop in reward was common to other attempts to train Qwen models with semantic mutual information reward as well. We hypothesize that this occured because the Qwen model has a narrow distribution of possible sentence completions compared to the other two models. Thus, when the semantic reward term was included, the Qwen model attempted to diversify its output for some strategy, but had no more sensible English completions not already used by other strategies, resulting in the remaining strategies becoming nonsensical. Further investigation is needed to determine the veracity this claim.

Finally, one observation apparent in all three training graphs in Figure 7 is that the semantic mutual information reward seems to be highly correlated with the correctness reward. This is surprising because the output diversity intuitively has a *negative* correlation with correctness, as we expect each strategy to be correct less of the time, in exchange for at least one of the strategies being correct more of the time. Further investigation is required to figure out what may have induced this correlation - whether it is because the embedding model and/or KSG estimator are somehow biased towards correct outputs, or if output diversity and correctness are linked in some other way.

6 Discussion and Limitations

Due to limitations in compute, our training was more limited than we would have preferred, and we hope to preform more experiments in the future to further validate our results.

In the preferred case, rather than training on middle school difficulty problems that often have only a single broad way to approach the answer, we would have leveraged significantly larger models (such as Qwen3 [13] with 235B parameters or Llama 4 Maverick [12] with 400B parameters) and much harder problems (such as in U-MATH, DeepMath, or FrontierMath [1] [6] [4]). These harder datasets contains problems that have many distinct possible approaches of, say, algebra, geometry, combinatorics, etc., which is where we expect output diversity to be the most effective.

For future work, we wish to verify whether the distinct strategies that were learned actually provide a new perspective semantically. For example, in certain cases, the model aims to maximize the mutual information reward by changing formatting attributes or even speaking less intelligibly. Although this does change the output distribution, these are not desirable strategies to increase model performance. We hope to work on techniques that prioritize semantic diversity over token-level diversity, and investigate to what extent our current techniques do so.

Acknowledgements. We thank Prof. Eysenbach and Chongyi Zheng for the original MISL idea and paper. We also thank the TAs in COS 435 for having such prompt response times and being helpful in developing our project.

References

[1] Chernyshev, K., Polshkov, V., Artemova, E., Myasnikov, A., Stepanov, V., Miasnikov, A., and Tilga, S. (2024). U-math: A university-level benchmark for evaluating mathematical skills in llms. *arXiv preprint arXiv:2412.03205*.

[2] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

[3] Daniel Han, M. H. and team, U. (2023). Unsloth.

[4] Glazer, E., Erdil, E., Besiroglu, T., Chicharro, D., Chen, E., Gunning, A., Olsson, C. F., Denain, J.-S., Ho, A., Santos, E. d. O., et al. (2024). Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. *arXiv preprint arXiv:2411.04872*.

[5] Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

[6] He, Z., Liang, T., Xu, J., Liu, Q., Chen, X., Wang, Y., Song, L., Yu, D., Liang, Z., Wang, W., et al. (2025). Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*.

[7] Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al. (2024). Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

[8] Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. (2024). Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

[9] Kraskov, A., Stoegbauer, H., and Grassberger, P. (2004). Estimating Mutual Information. *Physical Review E*, 69(6):066138. arXiv:cond-mat/0305641.

[10] Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., et al. (2022). Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

[11] Lyu, Z., Li, X., Xie, Z., and Li, M. (2025). Top pass: improve code generation by pass@ k-maximized code ranking. *Frontiers of Computer Science*, 19(8):198341.

[12] Meta AI (2025). The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. Meta AI Blog. Announcement of the Llama 4 multimodal AI model family. Accessed: August 7, 2025.

[13] Qwen Team (2025). Qwen3: Think deeper, act faster. Qwen Blog. Announcement of Qwen3 large language model family. Accessed: August 7, 2025.

[14] Shaier, S., Sanz-Guerrero, M., and Wense, K. v. d. (2025). Asking Again and Again: Exploring LLM Robustness to Repeated Questions. arXiv:2412.07923 [cs].

[15] Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. (2024). Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

[16] Shur-Ofry, M., Horowitz-Amsalem, B., Rahamim, A., and Belinkov, Y. (2024). Growing a Tail: Increasing Output Diversity in Large Language Models. arXiv:2411.02989 [cs].

[17] Steeg, G. V. (2025). gregversteeg/npeet. original-date: 2014-10-10T19:57:02Z.

[18] Trinh, T. H., Wu, Y., Le, Q. V., He, H., and Luong, T. (2024). Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482.

[19] Unsloth.ai (2025). Llama3.1 (8b)-grpo notebook. `https://colab.research.google.com/github/unslothai/notebooks/blob/main/nb/Llama3.1`.

[20] Zheng, C., Tuyls, J., Peng, J., and Eysenbach, B. (2024). Can a misl fly? analysis and ingredients for mutual information skill learning. *arXiv preprint arXiv:2412.08021*.

A KSG Estimator for Mutual Information

The basic idea of the KSG estimation method for mutual information is as follows: for $X \subset \mathbb{R}^{d_x}, Y \subset \mathbb{R}^{d_y}$ with $|X| = |Y| = n$, suppose we wish to approximate $\mathcal{I}(X,Y) = \mathcal{H}(X) + \mathcal{H}(Y) - \mathcal{H}(X,Y)$. Let Z be the joint distribution of (X,Y). We can then estimate the local mutual information as follows: for each point $z_i = (x_i, y_i)$ we draw the smallest square centered at z_i containing its kth nearest neighbor (where distances are computed using any norm over X and Y) - suppose that this square has length $2\epsilon_i$. Define $n_x(i) = \#\{j \mid ||x_j - x_i|| \leq \epsilon_i, j \neq i\}$ and analogously define $n_y(i)$. Then making the approximation that the density locally in this square is uniform, the interval in X of size $2\epsilon_i$ contains $n_x(i)+1$ points total out of N, so

$$\mathcal{H}(X) = -\mathbb{E}[\log p(X)] \approx -\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{n_x(i)+1}{2\epsilon_i n}\right), \mathcal{H}(Y) \approx -\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{n_y(i)+1}{2\epsilon_i n}\right)$$

and similarly since the square has area $4\epsilon_i^2$ that

$$\mathcal{H}(X,Y) = \mathcal{H}(Z) \approx -\sum_{i=1}^{n} \frac{1}{n} \log\left(\frac{k+1}{4\epsilon_i^2 n}\right).$$

Finally, to estimate the overall mutual information, note that several of these terms cancel, so we are left with $\mathcal{I}(X,Y) \approx \log(k+1) + \log n - \mathbb{E}_{1\leq i\leq n}[\log(n_x(i)+1) + \log(n_y(i)+1)]$. For the actual KSG estimator, instead of directly computing the expected value of logarithms, the digamma function is used to minimize the error in the most precise approximation [9].

B Additional Figures and Tables

This section contains additional figures or tables complementing the main report.

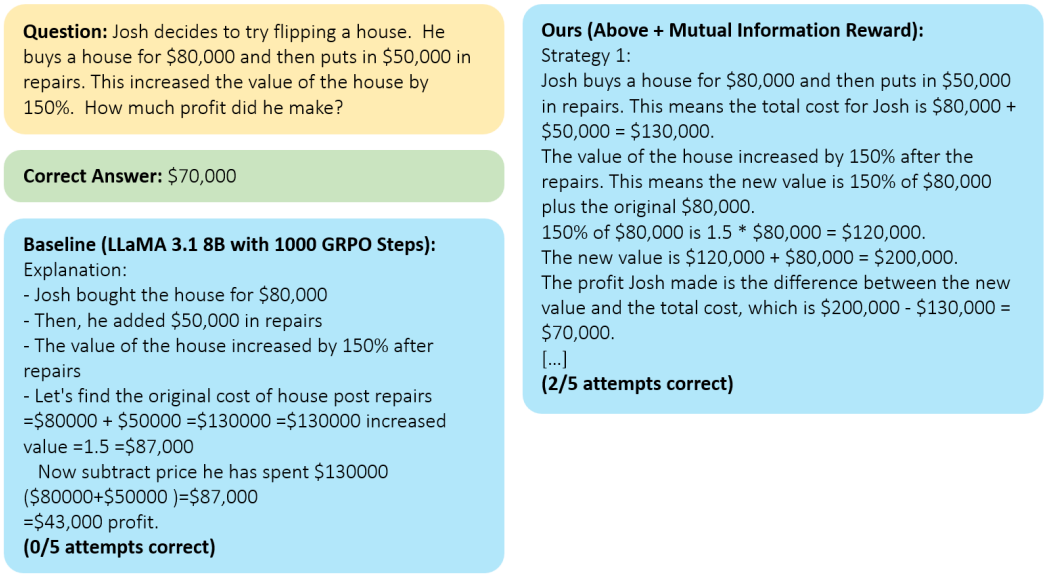

Figure 6: Comparison of example output of our model and baseline model with a question from the GSM8K dataset.

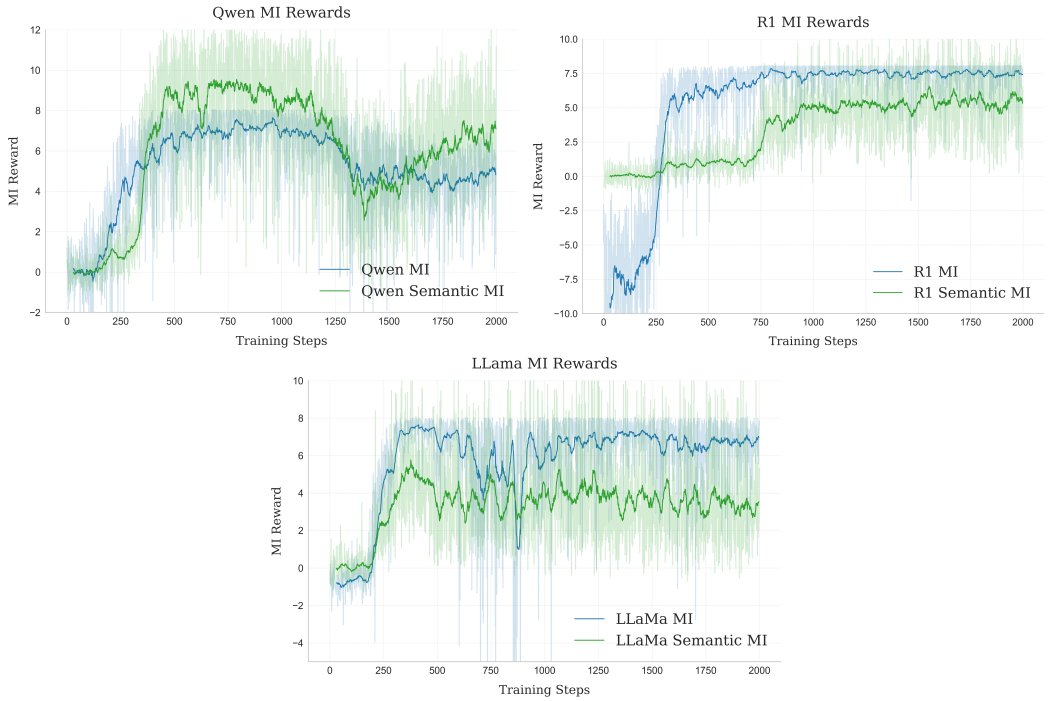

Figure 7: Mutual information reward in training for parameter regime $(\alpha_1, \alpha_2) = (5, 5)$ and 2000 training steps. Here, the blue line (labeled MI) represents the *token* mutual information reward.

Table 2: Performance statistics for (Model, α_1, α_2) configurations on 100 test problems, with $z = 5$ and $k = 5$.

Model	α_1	α_2	pass@1	pass@k	plu@k	cons@k
Llama	0	0	63	88	64	63
	1	1	72	90	83	67
	1	3	71	91	80	69
	2	2	**78**	**95**	76	63
	3	1	72	86	75	50
	5	0	75	86	**78**	**75**
	5	5	67	87	71	59
R1	0	0	33	70	46	19
	0.5	1.5	**39**	74	48	**27**
	1	1	38	75	44	21
	1	3	31	**82**	**50**	25
	5	0	**39**	80	44	23
	5	5	36	76	44	20
Qwen	0	0	51	83	71	56
	1	0.5	7	37	11	2
	1	1	20	51	24	1
	1	3	16	57	16	4
	2	2	4	20	3	0
	5	0	**73**	**87**	**74**	**69**
	5	5	3	33	8	0