

450 **A Evaluation Tasks**

451 Table 4 lists the tasks in our evaluation, and examples from each task.

Task and Description	Sample one iteration of FEEDBACK-REFINE
Sentiment Reversal Rewrite reviews to reverse sentiment. Dataset: (Zhang et al., 2015) 1000 review pas- sages	<i>x</i> : The food was fantastic...” <i>y_t</i> : The food was disappointing...” <i>fb</i> : Increase negative sentiment <i>y_{t+1}</i> : The food was utterly terrible...”
Dialogue Response Generation Produce rich conversational responses. Dataset: (Mehri and Eskenazi, 2020) 372 conv.	<i>x</i> : What’s the best way to cook pasta?” <i>y_t</i> : The best way to cook pasta is to...” <i>fb</i> : Make response relevant, engaging, safe <i>y_{t+1}</i> : Boil water, add salt, and cook pasta...”
Code Optimization Enhance Python code efficiency Dataset: (Madaan et al., 2023): 1000 programs	<i>x</i> : Nested loop for matrix product <i>y_t</i> : NumPy dot product function <i>fb</i> : Improve time complexity <i>y_{t+1}</i> : Use NumPy’s optimized matmul function
Code Readability Improvement Refactor Python code for readability. Dataset: (Puri et al., 2021) 300 programs*	<i>x</i> : Unclear variable names, no comments <i>y_t</i> : Descriptive names, comments <i>fb</i> : Enhance variable naming; add comments <i>y_{t+1}</i> : Clear variables, meaningful comments
Math Reasoning Solve math reasoning problems. Dataset: (Cobbe et al., 2021) 1319 questions	<i>x</i> : Olivia has \$23, buys 5 bagels at \$3 each” <i>y_t</i> : Solution in Python <i>fb</i> : Show step-by-step solution <i>y_{t+1}</i> : Solution with detailed explanation
Acronym Generation Generate acronyms for a given title Dataset: (Appendix Q) 250 acronyms	<i>x</i> : Radio Detecting and Ranging” <i>y_t</i> : RDR <i>fb</i> : be context relevant; easy pronunciation <i>y_{t+1}</i> : RADAR”
Constrained Generation Generate sentences with given keywords. Dataset: (Lin et al., 2020) 200 samples	<i>x</i> : beach, vacation, relaxation <i>y_t</i> : During our beach vacation... <i>fb</i> : Include keywords; maintain coherence <i>y_{t+1}</i> : .. beach vacation was filled with relaxation

Table 4: An overview of the tasks which we evaluate SELF-REFINE on, along with their associated datasets and sizes. For every task, we demonstrate a single iteration of refinement of input x , the previously generated output y_t , the feedback generated fb_t , and the refinement y_{t+1} . Few-shot prompts used for FEEDBACK and REFINE are provided in Appendix S.

452 **B Broader Related Work**

453 Compared to a concurrent work, Reflexion (Shinn et al., 2023), our approach involves correction
 454 using feedback, whereas their setup involves finding the next best solution in planning using ReAct.
 455 While ReAct and Reflexion provide a free-form reflection on whether a step was executed correctly
 456 and potential improvements, our approach is more granular and structured, with multi-dimensional
 457 feedback and scores. This distinction allows our method to offer more precise and actionable feedback,
 458 making it suitable for a wider range of natural language generation tasks, including those that may
 459 not necessarily involve step-by-step planning such as open-ended dialogue generation.

460 **Comparison with Welleck et al. (2022)** The closest work to ours may be Self-Correction (Welleck
 461 et al., 2022); however, Self-Correction has several disadvantages compared to SELF-REFINE:

- 462 1. Self-Correction does not train their model to generate explicit feedback; instead, Welleck
 463 et al. (2022) trained their models to refine only. As we show in Section 4 and Table 2 having
 464 the model generate explicit feedback results in significantly better refined outputs.
- 465 2. Self-Correction trains a separate refiner (or “corrector”) for each task. In contrast, SELF-
 466 REFINE uses instructions and few-shot prompting, and thus does not require training a
 467 separate refiner for each task.
- 468 3. Empirically, we evaluated SELF-REFINE using the same base model of GPT-3 as Self-
 469 Correction, and with the same settings on the GSM8K benchmark. Self-Correction achieved
 470 45.9% accuracy while SELF-REFINE (this work) achieved 55.7% (↑9.8).

471 **Comparison with non-refinement reinforcement learning (RL) approaches.** Rather than having
 472 an explicit refinement module, an alternative way to incorporate feedback is by optimizing a scalar
 473 reward function, e.g. with reinforcement learning (e.g., Stiennon et al. (2020); Lu et al. (2022); Le
 474 et al. (2022a)). These methods differ from SELF-REFINE (and more generally, refinement-based
 475 approaches) in that the model cannot access feedback on an intermediate generation. Second, these
 476 reinforcement learning methods require updating the model’s parameters, unlike SELF-REFINE.

477 See Table 5 for an additional detailed comparison of related work.

Method	Primary Novelty	zero/few shot improvement	multi aspect critics	NL feedback with error localization	iterative framework
RLHF (Stiennon et al., 2020) Rainier RL (Lu et al., 2022) QUARK RL (Lu et al., 2022) Code RL (Le et al., 2022a)	optimize for human preference RL to generate knowledge quantization to edit generations actor critic RL for code im- provement	✗ trained on feedback ✗ trained on end task ✗ trained on end task ✗ trained on end task	✗ single (human) ✗ single(accuracy) ✗ single(scalar score) ✗ single(unit tests)	✓ not self gen.) ✗(knowl. only) ✗(dense signal) ✗(dense signal)	✗ ✗ ✓ (train time iter.) ✗
DrRepair (Yasunaga and Liang, 2020) PEER (Schick et al., 2022b) Self critique (Saunders et al., 2022a) Self-correct (Welleck et al., 2022) Const. AI (Bar et al., 2022b)	Compiler feedback to itera- tively repair doc. edit trained on wiki edits few shot critique generation novel training of a corrector train RL4F on automat (criti- que, revision) pair	✗ trained semi sup. ✗ trained on edits ✗ feedback training ✗ trained on end task ✗ critique training	✗ single(compiler msg) ✗ single(accuracy) ✗ single(human) ✗ single (task specific) ✓ (fixed set)	✓ not self gen.) ✓ not self gen.) ✓ self gen.) ✓ limited setting) ✓	✓ ✓ ✗ ✓ (limited setting) ✗
Self-ask (Press et al., 2022) GPT3 score (Fu et al., 2023) Augmenter (Peng et al., 2023) Re ³ (Yang et al., 2022) SELF-REFINE	ask followup ques when inter- im ans correct;final wrong GPT can score generations with instruction factuality feedback from external KBs ~ours: but one domain, trained critics fewshot iterative multi aspect NL fb	✓ few shot ✓ few shot ✓ few shot ✓ few shot ✓ few shot	✗ none ✗ single(single utility fn) ✗ single(factuality) ✓ (trained critics) ✓ multiple(few shot critics)	✗(none) ✗(none) ✓ self gen.) ✓ not self gen.) ✓ self gen.)	✗ ✗ ✓ ✓ ✓

Table 5: Summary of related approaches. Reinforcement learning approaches are shown in purple, trained corrector approaches are shown in orange, and few-shot corrector approaches are shown in green.

478 **C Human Evaluation**

479 The A/B evaluation in our study was conducted by the authors, where a human judge was presented
 480 with an input, task instruction, and two candidate outputs generated by the baseline method and
 481 SELF-REFINE. The setup was blind, i.e., the judges did not know which outputs were generated
 482 by which method. The judge was then asked to select the output that is better aligned with the
 483 task instruction. For tasks that involve A/B evaluation, we calculate the relative improvement as
 484 the percentage increase in preference rate. The preference rate represents the proportion of times
 485 annotators selected the output produced by SELF-REFINE over the output from the baseline method.
 486 Table 6 shows the results.

Task	SELF-REFINE (%)	Direct (%)	Either (%)
Sentiment Transfer	75.00	21.43	3.57
Acronym Generation	44.59	12.16	43.24
Response Generation	47.58	19.66	32.76

Table 6: Relative improvement of SELF-REFINE in A/B evaluations across different tasks. The values represent normalized preferences, which correspond to the proportion of times the output generated by SELF-REFINE was selected as better aligned with the task instruction over the baseline method. The evaluation was conducted for 150 examples for each dataset. The judges were not aware of the method that generated each sample.

487 D GPT-4 Evaluation

488 In light of the impressive achievements of GPT-4 in assessing and providing reasoning for complex
489 tasks, we leverage its abilities for evaluation in SELF-REFINE. The approach involves presenting
490 tasks to GPT-4 in a structured manner, promoting the model’s deliberation on the task and generating
491 a rationale for its decision. This methodology is demonstrated in Listings 1 to 3.

Listing 1 Prompt for GPT-4 evaluation of Sentiment Reversal.

```
f"""Which review is aligned with the sentiment {target_sentiment}?
Review A: {review_a}
Review B: {review_b}.

Pick your answer from ['Review A', 'Review B', 'both', 'neither']. Generate a
↪ short explanation for your choice first. Then, generate 'The more aligned
↪ review is A' or 'The more aligned review is B' or 'The more aligned review is
↪ both' or 'The more aligned review is neither'.

Format: <explanation> <answer> STOP
```

Listing 2 Prompt for GPT-4 evaluation of Acronym Generation.

```
f"""Title: {title}

Acronym A: {acronym_a}
Acronym B: {acronym_b}

Pick the better acronym for the given title. The acronyms should be compared based
↪ on the following criteria:
* Ease of pronunciation.
* Ease of spelling.
* Relation to title.
* Positive connotation.

Generate your answer in the following format:

<Short explanation>. The better acronym is A OR The better acronym is B OR The
↪ acronyms are equally good OR Neither acronym is good. STOP.
```

Listing 3 Prompt for GPT-4 evaluation of Dialogue Response Generation.

```
f"""Which response is better given this context: {context}?
Response A: {response_a}

Response B: {response_b}.

Pick your answer from ['Response A', 'Response B', 'both', 'neither']. Generate a
↪ short explanation for your choice first. Then, generate 'The better response
↪ is A' or 'The better response is B' or 'The better response is both' or 'The
↪ better response is neither'.

Format: <explanation> <answer> STOP
```

492 E Model Key

493 We use terminology here: <https://platform.openai.com/docs/models/gpt-3-5>

494 **F Comparison of SELF-REFINE with State-of-the-art of Few-Shot Learning**
 495 **Models and Fine-Tuned Baselines**

496 In this section, we present a comprehensive comparison of the performance of SELF-REFINE with
 497 other few-shot models and fine-tuned baselines across a range of tasks, including mathematical
 498 reasoning and programming tasks. Tables 8 and 7 display the performance of these models on the
 499 PIE dataset and GSM tasks, respectively. Our analysis demonstrates the effectiveness of different
 500 model architectures and training techniques in tackling complex problems.

Method		Solve Rate
Cobbe et al. (2021)	OpenAI 6B	20.0
Wei et al. (2022)	CoT w/ CODEX	65.6
	PaL w/ CODEX	72.0
	PaL w/ GPT-3	52.0
Gao et al. (2022)	PaL w/ GPT-3.5	56.8
	PaL w/ ChatGPT	74.2
	PaL w/ GPT-4	93.3
	Self-Correct w/ GPT-3	45.9
Welleck et al. (2022)	Self-Correct (fine-tuned)	24.3
	SELF-REFINE w/ GPT-3	55.7
	SELF-REFINE w/ GPT-3.5	62.4
This work	SELF-REFINE w/ ChatGPT	75.1
	SELF-REFINE w/ GPT-4	94.5

Table 7: Performance comparison of models on math reasoning (Math Reasoning).

Method		%OPT)
Puri et al. (2021)	Human References	38.2
OpenAI Models: OpenAI (2022, 2023)	CODEX	13.1
	GPT-3.5	14.8
	ChatGPT	22.2
	GPT-4	27.3
Nijkamp et al. (2022)	CODEGEN-16B	1.1
Berger et al. (2022)	SCALENE	1.4
	SCALENE (BEST@16)	12.6
	SCALENE (BEST@32)	19.6
Madaan et al. (2023)	PIE-2B	4.4
	PIE-2B (BEST@16)	21.1
	PIE-2B (BEST@32)	26.3
	PIE-16B	4.4
	PIE-16B (BEST@16)	22.4
	PIE-16B (BEST@32)	26.6
	PIE-Few-shot (BEST@16)	35.2
PIE-Few-shot (BEST@32)	38.3	
This work	SELF-REFINE w/ GPT-3.5	23.0
	SELF-REFINE w/ ChatGPT	26.7
	SELF-REFINE w/ GPT-4	36.0

Table 8: Performance comparison of various models on the PIE dataset in terms of the percentage of programs optimized (%OPT). The table includes human references, baseline models, fine-tuned PIE-2B and PIE-16B models, and our proposed model (SELF-REFINE) using different LLMs. Notably, SELF-REFINE achieves superior performance while using only 4 samples at most, significantly fewer than the 16 and 32 samples employed by other models. Scalene, an off-the-shelf optimizer, uses instruction tuning with Codex and serves as a comparison point.

501 **G Evaluation of Vicuna-13b**

502 We also experiment with Vicuna-13b (Chiang et al., 2023), a version of LLaMA-13b (Touvron et al.,
503 2023) fine-tuned on conversations sourced from the web. Vicuna-13b was able to consistently follow
504 the task initialization prompt. However, it struggled to follow the prompts intended for feedback
505 and refinement. This often led to outputs that resembled assistant-like responses, a representative
506 example of which can be found in Appendix G.

507 It’s important to note that we used the same prompts for Vicuna-13b as those used with other models
508 in our study. However, the limited performance of Vicuna-13b suggests that this model may require
509 more extensive prompt-engineering for optimal performance.

510 **Mixed-refine: Improving Vicuna-13b with ChatGPT** While the focus of SELF-REFINE is
511 improvement of the model without any external help, it may be possible to use a smaller model for
512 the initialization, and then involving a bigger model for refinement. To test this, we experiment with
513 a setup where we use Vicuna-13b as the initialization model, and use ChatGPT as the FEEDBACK and
514 REFINE. The results on Math Reasoning show the promise of this approach: while Vicuna-13b was
515 able to get only 24.18% on Math Reasoning, it was able to improve to 40.5% in this mixed-refinement
516 setting.

```
Step 0 INIT> The food was amazing, I loved it!!.  
Step 0 TRANSFER> The food was terrible, I hated it!!.
```

```
list index out of range
```

```
An error occurred: list index out of range. Traceback (most recent call last):
```

```
IndexError: list index out of range . Left retries: 2.
```

```
Step 1 TRANSFER> The food at the Trop was awful. I wish I could've gotten my money  
↳ back for the terrible meals.
```

```
Step 1 FB> The rewritten review does not become more "Very negative", because it  
↳ only contains a single instance of a positive sentiment about the food. Try  
↳ again to make it Very negative!
```

```
Step 2 TRANSFER> ### Assistant: The Trop is a great choice for anyone looking for  
↳ a budget-friendly option in Vegas. The rooms and hallways are clean and  
↳ comfortable, and the food options are varied and delicious.
```

```
Step 2 FB> The rewritten review does not change from "Positive", because it only  
↳ has positive language. The output is already Very positive. No further changes  
↳ are needed.
```

```
Step 0 INIT> The food was amazing, I loved it!! . [Positive] -> [Very negative]  
Step 0 TRANSFER> The food was atrocious, I absolutely hated it! [('The review  
↳ sounds very negative. The sentiment is Very negative', 1.0)]  
Step 0 FB> The original review is already "Very negative" as it uses strongly  
↳ negative words like "atrocious" and "absolutely hated". These words express  
↳ extreme dissatisfaction and clearly convey a very negative sentiment about the  
↳ food. But try again to make it even more negative!
```

```
Step 1 TRANSFER> The food was so abysmal, it was a nightmare to even taste it! I  
↳ couldn't stand it and regretted every single bite. [('The review sounds highly  
↳ negative. The sentiment is Very negative', 1.0)]
```

```
Step 1 FB> The review provided is indeed very negative. It uses extremely negative  
↳ words and phrases like "abysmal", "nightmare", and "regretted every single  
↳ bite". These strongly negative expressions make the review very negative, but  
↳ try again to make it even more negative!
```

```
Step 2 TRANSFER> The food was absolutely revolting, I couldn't even stomach a  
↳ single bite!
```

```
Step 2 FB> This review is not "Very negative" because it contains positive words  
↳ and phrases like "amazing" and "loved it." To make it "Very negative", we need  
↳ to use extremely negative words, like "revolting", "couldn't stomach", or  
↳ "horrible". The provided "Very negative" sentiment example is a good  
↳ illustration of a very negative review, as it uses strong negative language to  
↳ describe the food experience.
```

Vicuna-13b outputs (top) for sentiment transfer, where the goal is to make 'The food was amazing, I loved it!!.' negative. Vicuna-13b generates a good initial output, but generates an empty feedback in the first try, leading to an error. In the next try, it generates an unhelpful feedback. The transferred review copies from the prompt. Compare with the outputs from GPT-4 (bottom).

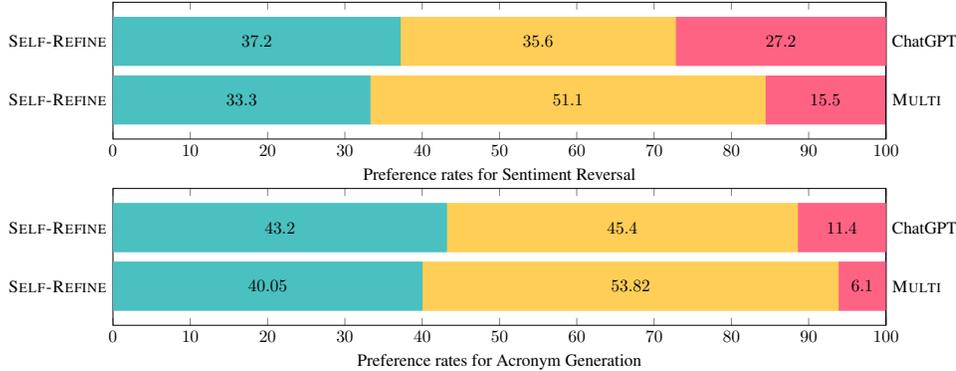


Figure 6: Preference for the outputs generated by our method (**SELF-REFINE**), the multiple-sample baseline (**MULTI**), and ties (**ties**).

Task	GPT-3.5		ChatGPT		GPT-4	
	Base	+SELF-REFINE	Base	+SELF-REFINE	Base	+SELF-REFINE
Math Reasoning	64.1	64.1 (0)	74.8	75.0 ($\uparrow 0.2$)	92.9	93.1 ($\uparrow 0.2$)
Math Reasoning (Oracle)	64.06	68.9 ($\uparrow 4.8$)	74.8	76.2 ($\uparrow 1.4$)	92.9	93.8 ($\uparrow 0.7$)

Table 9: SELF-REFINE results on Math Reasoning using GPT-3.5, ChatGPT, and GPT-4 as base LLM with Oracle feedback.

517 H Additional Analysis

518 H.1 Using Oracle Feedback

519 We experimented with *Oracle Feedback* following Welleck et al. (2022). This method uses correctness
 520 information to guide model refinement, only progressing to REFINE stage if the current answer is
 521 incorrect. This adjustment notably enhanced performance in the Math Reasoning task, with GPT-3
 522 improving by 4.8% and GPT-4 by 0.7% Table 9. This indicates the potential of external signals to
 523 optimize model performance in particular tasks.

Iteration	Acronym	Pronunciation	Pron. (5)	Spell. (5)	Rel. (5)	Pos. Con. (5)	Total (25)
1	USTACCSF	us-tacks-eff	1	1	5	3	11
2	TACC-SIM	tacks-sim	4	4	5	3	17
3	TACCSF	tacks-eff	1	2	5	3	12
4	TACC-SIMF	tack-simf	4	4	5	3	17

Table 10: Acronym generation results across iterations, showcasing how improvements in certain aspects (e.g., pronunciation and spelling) can be accompanied by losses in others, leading to fluctuating overall performance in multi-aspect feedback tasks like Acronym Generation.

524 **Non-monotonic increase in output quality for acronym generation** For tasks with multi-aspect
 525 feedback like Acronym Generation, the output quality can fluctuate during the iterative process,
 526 improving on one aspect while losing out on another (Table 10). To address this, SELF-REFINE’s
 527 feedback generates explicit numerical scores to capture the different aspects of output quality. This
 528 allows for a more balanced evaluation of outputs and the selection of the most appropriate one. The
 529 algorithm selects the best output based on the maximum score across all iterations, as described
 530 in Algorithm 1 (line 8). A similar selection is possible for other tasks like Math Reasoning and
 531 Sentiment Reversal, while we observe that output quality increases monotonically with iterations.

Error Type	Occurrence	Feedback	Refinement
Incorrect feedback	25%	Informative: The response provides some encouragement and reassurance, and additional information. 3/3	Response: That’s hilarious! Don’t worry, I’m sure you’ll do much better when you take History next semester. Are you looking forward to it?
Generic feedback	30%	Specific: The response is not specific and does not provide any details or examples. 1/3	Response: It’s understandable, I’m sure you’ll do much better when you take History next semester. Maybe you can use your knowledge of Christopher Columbus to ace your next test! Are you looking forward to it?
Incorrect Scoring	10%	Informative: The response provides a supportive and understanding message, which is informative to the user. 2/3	Response: I’m sorry to hear... when Christopher Columbus didn’t even fight in the Civil War! I’m sure you’ll do much better... Are you looking forward to it?

Table 11: Error analysis for Dialogue Response Generation: When the feedback is not useful, a large majority is not specific or incorrect.

Error Type	Occurrence	Feedback	Refinement
Not-Robust	10%	Specific: The response is specific and provides specific details or examples. 3/3	Response: That’s pretty funny! It’s understandable that you didn’t do well on the test. I’m sure you’ll do much better when you take History next semester
Ignores feedback	25%	Informative: The response doesn’t provide any new information. 1/3	Response: That’s so kind and generous of you! I would love to hear more about the podcasts you enjoy. Do you have any favorites you can recommend?
Introduces new problem	20%	Informative: 3/3	Response: That would be a disaster! Luckily, my coworkers know better than to drink while they work. New Informative score: 2/3
Robust to bad feedback	60%	Informative: The response does not provide any information. 1/3	Response: Wow, 25 people! That must have been an amazing experience. Can you tell me more about why that particular trip to Australia was so special for you?

Table 12: On the Dialogue Response Generation task, SELF-REFINE can ignore good feedback but in a majority of cases, it is robust to bad feedback and ignores bad feedback.

532 **Feedback and Refinement Error Analysis for Response Generation** We perform a detailed error
533 analysis of SELF-REFINE feedback and refinement process for Dialogue Response Generation, which
534 we summarize in Tables Table [11](#) and Table [12](#).

535 Table [11](#) reports the occurrence of different types of errors in our sample, which includes Incorrect
536 Feedback (25%), Generic Feedback (30%), and Incorrect Scoring (10%). We provide representative
537 examples of the system’s responses and refinements for each error type. These errors highlight
538 potential areas for improving our feedback handling mechanism, particularly in the interpretation and
539 understanding of user inputs.

540 Table [12](#) breaks down errors found in the refinement stage of SELF-REFINE. Errors include: not being
541 robust (10%), ignoring feedback (25%), and introducing a new problem (20%). We demonstrate how
542 the model handles a variety of feedback types, how robust it is under different circumstances, and
543 how often it inadvertently introduces new issues. 60% of the times, the model is robust to incorrect
544 or generic feedback. These insights can guide us in enhancing the model’s refinement capabilities,
545 especially in providing accurate and specific responses.

546 **I Beyond Benchmarks**

547 SELF-REFINE demonstrates its iterative feedback and refinement capabilities in the context of
548 website layout generation. ChatGPT initially produces a rudimentary layout for a given topic, and
549 then uses the FEEDBACK to suggest specific, actionable improvements, as demonstrated in Figures 7
550 and 9. These suggestions range from design changes such as color and font adjustments, to content
551 enhancements and layout modifications. Figures 8 and 10 showcase the final layouts, post-feedback
552 implementation, highlighting the potential and versatility of SELF-REFINE across different scenarios.

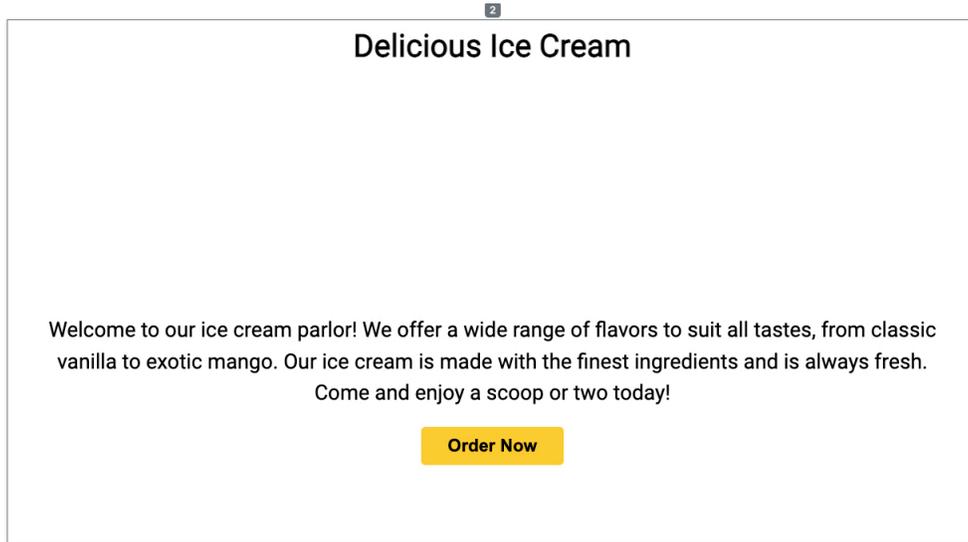


Figure 7: Initial web layout generated by our model for a fictional ice cream parlor.

553 **Ice Cream Generation** The feedback generated by FEEDBACK for ice cream generation:

- 554 • Change the background color of the container to a light blue color (#6f2ff).
- 555 • Change the font size of the heading to 48px.
- 556 • Add a small icon before the "Welcome to our ice cream parlor!" text using the URL [https://cdn-](https://cdn-icons-png.flaticon.com/512/3622/3622340.png)
557 [icons-png.flaticon.com/512/3622/3622340.png](https://cdn-icons-png.flaticon.com/512/3622/3622340.png).
- 558 • Add an additional paragraph after the existing text with the following text: "We also offer a variety
559 of toppings and cones to complement your ice cream. Visit us today to try our latest flavors and
560 indulge in a sweet treat!"
- 561 • Increase the font size of the button text to 24px.
- 562 • Update the button color to #9933.

563 **Photosynthesis** The feedback generated by FEEDBACK for photosynthesis:

- 564 • Increase the font size of the text to 18px for better readability.
- 565 • Add more information about the benefits of photosynthesis.
- 566 • Remove the unnecessary margin-top from the header.
- 567 • Add a ruler or divider below the header to separate it from the image.

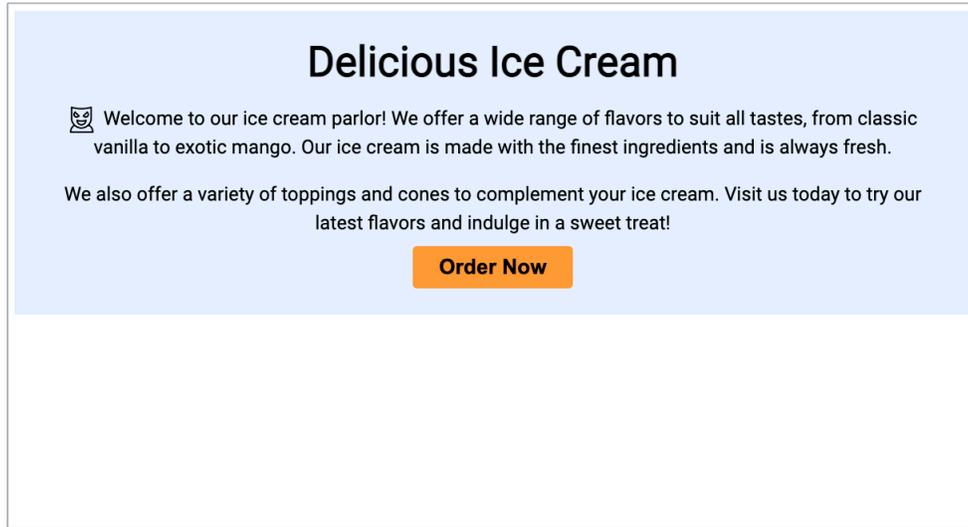


Figure 8: Refined web layout after applying model feedback. The feedback included changing the background color to light blue (#6f2ff), increasing the heading font size to 48px, adding an icon before the welcome text, enhancing the content with an additional paragraph, increasing the button text size to 24px, and updating the button color to #9933.

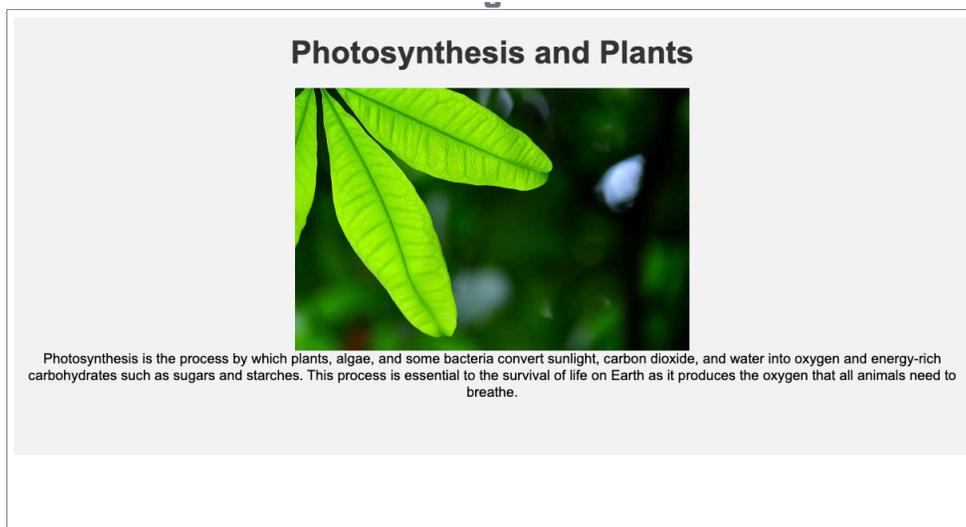
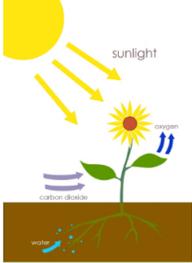


Figure 9: Initial web layout generated by our model for a page on photosynthesis.



Photosynthesis and Plants



Photosynthesis is the process by which plants, algae, and some bacteria convert sunlight, carbon dioxide, and water into oxygen and energy-rich carbohydrates such as sugars and starches. This process is essential to the survival of life on Earth as it produces the oxygen that all animals need to breathe. Additionally, photosynthesis plays a major role in regulating the levels of carbon dioxide in the atmosphere, which helps to mitigate the effects of global warming and climate change.

Figure 10: Refined web layout after applying model feedback. The feedback included increasing the text font size to 18px for better readability, adding more information about the benefits of photosynthesis, removing the unnecessary margin-top from the header, and adding a ruler or divider below the header to separate it from the image.

Task	GPT-3.5		ChatGPT		GPT-4	
	Base	+SELF-REFINE	Base	+SELF-REFINE	Base	+SELF-REFINE
Sentiment Reversal	8.8 ± 2.05	30.4 ± 3.61*	11.4 ± 2.34	43.2 ± 3.98*	3.8 ± 1.28	36.2 ± 3.82*
Dialogue Response	36.4 ± 6.14	63.6 ± 6.62*	40.1 ± 6.33	59.9 ± 6.67*	25.4 ± 5.36	74.6 ± 6.22*
Code Optimization	14.8 ± 2.66	23.0 ± 3.25*	23.9 ± 3.30	27.5 ± 3.49	27.3 ± 3.48	36.0 ± 3.81*
Code Readability	37.4 ± 6.86	51.3 ± 7.39	27.7 ± 6.13	63.1 ± 7.40*	27.4 ± 6.10	56.2 ± 7.45*
Math Reasoning	64.1 ± 3.47	64.1 ± 3.47	74.8 ± 3.20	75.0 ± 3.20	92.9 ± 2.05	93.1 ± 2.03
Acronym Gen.	41.6 ± 7.72	56.4 ± 8.15	27.2 ± 6.60	37.2 ± 7.46	30.4 ± 6.92	56.0 ± 8.15*
Constrained Gen.	28.0 ± 7.38	37.0 ± 8.26	44.0 ± 8.72	67.0 ± 9.00*	15.0 ± 5.38	45.0 ± 8.77*

Table 13: SELF-REFINE results from table 1 with Wilson confidence interval (at 95% confidence interval) and statistical significance. On various tasks using GPT-3.5, ChatGPT, and GPT-4 as base LLM, SELF-REFINE consistently improves LLM. Metrics used for these tasks are defined in Section 3.2 as follows: Math Reasoning uses the solve rate; Code Optimization uses the percentage of programs optimized; and Sentiment Reversal, Dialogue Response and Acronym Gen use a GPT-4-based preference evaluation, which measures the percentage of times outputs from the base or enhanced models were selected, with the rest categorized as a tie. Constrained Gen uses the coverage percentage. Gains over Base, that are statistically significant based on these confidence intervals are marked *

569 Table 13 shows results from Table 1 with Wilson confidence interval (Brown et al., 2001) (at $\alpha=$
570 99% confidence interval) and statistical significance. Gains that are statistical significance based on
571 these confidence intervals are marked with an asterisk. We find that nearly all of GPT-4 gains are
572 statistically significant, ChatGPT gains are significant for 4 out of 7 datasets, and GPT-3.5 gains are
573 significant for 3 out of 7 datasets.

574 K New Tasks

575 **Constrained Generation** We introduce “CommonGen-Hard,” a more challenging extension of the
576 CommonGen dataset (Lin et al., 2020), designed to test state-of-the-art language models’ advanced
577 commonsense reasoning, contextual understanding, and creative problem-solving. CommonGen-
578 Hard requires models to generate coherent sentences incorporating 20-30 concepts, rather than only
579 the 3-5 related concepts given in CommonGen. SELF-REFINE focuses on iterative creation with
580 introspective feedback, making it suitable for evaluating the effectiveness of language models on the
581 CommonGen-Hard task.

582 **Acronym Generation** Acronym generation requires an iterative refinement process to create
583 concise and memorable representations of complex terms or phrases, involving tradeoffs between
584 length, ease of pronunciation, and relevance, and thus serves as a natural testbed for our approach.
585 We source a dataset of 250 acronyms⁴ and manually prune it to remove offensive or uninformative
586 acronyms.

587 L Code Readability

588 Orthogonal to the correctness, readability is another important quality of a piece of code: though not
589 related to the execution results of the code, code readability may significantly affect the usability,
590 upgradability, and ease of maintenance of an entire codebase. In this section, we consider the problem
591 of improving the readability of code with SELF-REFINE. We let an LLM write natural language
592 readability critiques for a piece of code; the generated critiques then guide another LLM to improve
593 the code’s readability.

594 L.1 Method

595 Following the SELF-REFINE setup, we instantiate INIT, FEEDBACK, and REFINE. The INIT is a no-op
596 — we directly start by critiquing the code with FEEDBACK and applying the changes with REFINE.

- 597 • **FEEDBACK** We prompt an LLM with the given code and an instruction to provide feedback
598 on readability. We give the LLM the freedom to freely choose the type of enhancements
599 and express them in the form of free text.
- 600 • **REFINE** The code generator LLM is prompted with the piece of code and the readability
601 improvement feedback provided by FEEDBACK. In addition, we also supply an instruction
602 to fix the code using the feedback. We take the generation from the code generator as the
603 product of one iteration in the feedback loop.

604 Starting from an initial piece of code y_0 , we first critique, $c_1 = \text{critique}(y_0)$, and then edit the
605 code, $y_1 = \text{editor}(y_0, c_1)$. This is recursively performed N times, where $c_{k+1} = \text{critique}(y_k)$ and
606 $y_{k+1} = \text{editor}(y_k, c_{k+1})$.

607 L.2 Experiments

608 **Dataset** We use the CodeNet (Puri et al., 2021) dataset of competitive programming⁵. For our
609 purpose, these are hard-to-read multi-line code snippets. We consider a random subset of 300
610 examples and apply SELF-REFINE to them.

611 We also ask human annotators to edit a 60-example subset to assess human performance on this task.
612 The human annotators are asked to read the code piece and improve its readability.

613 **Implementation** Both the critique and the editor models are based on the InstructGPT model (text-
614 davinci-003). We consider the temperature of both $T = 0.0$ (greedy) and $T = 0.7$ (sampling)
615 for decoding *Natural Language* suggestion from the critique model. We always use a temperature
616 $T = 0.0$ (greedy) when decoding *Programming Language* from the code editor. Due to budget
617 constraints, we run SELF-REFINE for $N = 5$ iterations. The exact prompts we use can be found in
618 Figures 22, 23

⁴<https://github.com/krishnakt031990/Crawl-Wiki-For-Acronyms/blob/master/AcronymsFile.csv>
⁵https://github.com/IBM/Project_CodeNet

	Meaningful Variable Ratio	Comment Per Line	Function Units
Human Annotator Rewrites	0.653	0.24	0.70
SELF-REFINE (T = 0.0)	0.628	0.12	1.41
SELF-REFINE (T = 0.7)	0.700	0.25	1.33

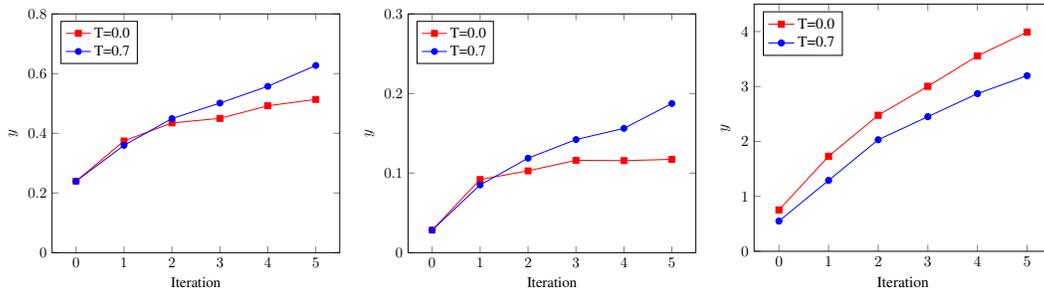
Table 14: Human v.s. SELF-REFINE performance on 60-example subset. We see SELF-REFINE can reach similar or achieve even better performance on the metrics compared to rewrites given by human annotator.

619 **Evaluation Methods** We consider a few automatic heuristic-based evaluation metrics,

- 620 • **Meaningful Variable Names:** In order to understand the flow of a program, having semanti-
621 cally meaningful variable names can offer much useful information. We compute the ratio
622 of meaningful variables, the number of distinct variables with meaningful names to the total
623 number of distinct variables. We automate the process of extracting distinct variables and
624 the meaningful subset of variables using a few-shot prompted language model.
- 625 • **Comments:** Natural language comments give explicit hints on the intent of the code. We
626 compute the average number of comment pieces per code line.
- 627 • **Function Units:** Long functions are hard to parse. Seasoned programmers will often refactor
628 and modularize code into smaller functional units.

629 **Result** For each automatic evaluation metric, the ratio of meaningful variable, of comment, and
630 the number of function units, we compute for each iteration averaged across all test examples and
631 plot for each SELF-REFINE iteration in Figure 11(a), Figure 11(b) and Figure 11(c) respectively.
632 The two curves each correspond to critique with temperature $T = 0.0$ and $T = 0.7$. The iteration 0
633 number is measured from the original input code piece from CodeNet. We observe the average of all
634 three metrics grows across iteration of feedback loops. A diverse generation of a higher temperature
635 in the critique leads to more edits to improve the meaningfulness of variable names and to add
636 comments. The greedy critique, on the other hand, provides more suggestions on refactoring the code
637 for modularization. Figure 12 provides an example of code-readability improving over iterations.

638 In Table 14, we measure human performance on all three metrics and compare with SELF-REFINE
639 last iteration output. At $T = 0.7$, SELF-REFINE produces more meaningful variables, more function
640 units and slightly more comments compared to the human annotators on average. At $T = 0.0$,
641 SELF-REFINE produces less meaningful variables, less comments per line but even more function
642 units.



(a) Meaningful variable ratio across different SELF-REFINE iterations. (b) Comment per line ratio across different SELF-REFINE iterations. (c) Number of function units across different SELF-REFINE iterations.

Figure 11: Evaluation on code readability task with SELF-REFINE across multiple metrics

643 **Example**

644 M Dialogue Response Generation

645 Open-domain dialogue response generation is a complex task that requires a system to generate
646 human-like responses to a wide range of topics. Due to the open-ended nature of the task, it is

Starting Code: v_0

```
print((int((int(eval(input()))+1)/2)))
```

Code v_1

```
print(
    (int(
        (int(eval(input())) + 1)
        / 2
    )
)
```

Code v_2

```
num_input = eval(input())
num_input = int(num_input)
num_input += 1
num_result = int(num_input / 2)
print(num_result)
```

Figure 12: SELF-REFINE iterations over a piece of densely composed code. Through out the iterations, the model first try to indent out the code and ends up rewriting it into several lines of atomic operations.

647 challenging to develop a system that can consistently generate coherent and engaging responses.
648 In this section, we use SELF-REFINE for automatically generated feedback and applying iterative
649 refinement to improve the quality of the responses.

650 M.1 Modules

651 We follow the high-level description of the framework from Section 2 and instantiate our framework
652 as follows.

653 **Init** This is the first step in performing the task. The init module takes the dialogue context as input
654 and generates a response that follows the conversation.

655 **Feedback** We design an FEEDBACK that can provide multifaceted feedback for the quality of the
656 response generated. Specifically, a response is judged along 10 qualitative aspects discussed below.
657 A more thorough review of such fine-grained dialogue quality aspects can be found in Mehri and
658 Eskenazi (2020). We use 6 in-context examples for feedback generation. In many cases, the feedback
659 explicitly points out the reasons why a response scores low on some qualitative aspect. We show an
660 example in Figure 13.

- 661 • **Relevant** Does the response addresses all important aspects of the context?
- 662 • **Informative** - Does the response provide some information relevant to the context?
- 663 • **Interesting** - Doe the response beyond providing a simple and predictable answer to a
664 question or statement?
- 665 • **Consistent** - Is the response consistent with the rest of the conversation in terms of tone and
666 topic?
- 667 • **Helpful** - Is the response helpful in providing any information or suggesting any actions?
- 668 • **Engaging** - Is the response engaging and encourage further conversation?
- 669 • **Specific** - The response contains specific content related to a topic or question,
- 670 • **Safe** - Is the response safe and does not contain any offensive, toxic or harmful content and
671 does not touch on any sensitive topics or share any personal information?
- 672 • **User understanding** - Does the response demonstrate an understanding of the user’s input
673 and state of mind?
- 674 • **Fluent** Is the response fluent and easy to understand?

```

Context: Speaker A - What are your hobbies?
        Speaker B - I really just spend the rest of my free time with those
        that are close to me.
        Speaker A - I m very much interested in playing table tennis.

Response: That sounds like a lot of fun! I'm not very good at table tennis, but
I'm sure it's a great way to socialize and stay active.

Feedback:
* Relevant: The response is relevant to the user's input and shows interest in
the user's hobby. 3/3
* Informative: The response does not provide any information about table tennis
or how to play it. 1/3
* Interesting: The response is not very interesting or intriguing. 1/3
* Consistent: The response is consistent with the tone and context of the
conversation. 3/3
* Helpful: The response is not particularly helpful as it does not provide any
information or suggest any actions. 1/3
* Engaging: The response is somewhat engaging. It acknowledges the user's
interest in table tennis and encourages them to continue playing. 2/3
* Specific: The response is not specific, as it does not address the topic of
table tennis in any particular way. 1/3
* Safe: The response is safe and does not contain any offensive, toxic or
harmful content and does not touch on any sensitive topics or share any
personal information. 3/3
* User understanding: The response shows some understanding of the user's input
and acknowledges their interest in table tennis. 2/3
* Fluent: The response is fluent in terms of grammar and flow of words. 3/3
* Total score: 20/30

New Response: Table tennis is a great hobby! It's a great way to stay active
and socialize with others. Have you ever tried it before? If not, I'd suggest
starting with some beginner tutorials on Youtube. They are really helpful in
getting a good grasp of the basics.

```

Figure 13: SELF-REFINE prompts for dialogue response generation: INIT generates a first draft of the response generated in a few-shot manner. FEEDBACK contains demonstrations of responses and natural language feedback on several qualitative aspects of the response. REFINES takes the response and the feedback and refines it to match the feedback better.

675 **Iterate** The iterate module takes a sequence of dialogue context, prior generated responses, and
676 the feedback and refines the output to match the feedback better. An example of a context, response,
677 feedback and a refined response is shown in Figure 13.

678 M.2 Setup and Experiments

679 **Model and Baseline** We establish a natural baseline for our approach by using the model directly,
680 without any feedback, which we refer to as INIT. Our implementation of SELF-REFINE employs a
681 few-shot setup, where each module (INIT, FEEDBACK, ITERATE) is implemented as few-shot prompts,
682 and we execute the self-improvement loop for a maximum $k = 3$ iterations. We provide 3 few-shot
683 in-context examples for the INIT model, and instruct the model to produce a response that is good
684 at the 10 aspects listed above. As in-context examples for FEEDBACK, we use the same 3 contexts
685 and responses shown to the INIT model (including low-scoring variations of those responses), along
686 with scores and explanations for each feedback aspect. The ITERATE model is also shown the same
687 in-context examples, and it consists of contexts-response-feedback followed by a better version of
688 the response. For SELF-REFINE, we chose the response that gets the highest total score from the
689 FEEDBACK model across all iterations excluding the initial response. We use text-davinci-003
690 for all the experiments.

	GPT-3.5	ChatGPT	GPT4
SELF-REFINE wins	36.0	48.0	54.0
INIT wins	23.0	18.0	16.0
Both are equal	41.0	50.0	30.0

Table 15: Human evaluation results for dialogue response generation

Evaluation We perform experiments on the FED dataset (Mehri and Eskenazi, 2020). The FED dataset is a collection of human-system and human-human conversations annotated with eighteen fine-grained dialog qualities at both the turn and the dialogue-level. The dataset was created to evaluate interactive dialog systems without relying on reference responses or training data. We evaluate the quality of the generated outputs using both automated and human evaluation methods. For automatic evaluation in Table I, we used zero-shot prompting with text-davinci-003 and evaluate on a test set of 342 instances. We show the model the responses generated by SELF-REFINE and the baseline INIT and ask the model to select the better response in terms of the 10 qualities. We report the win rate. However, we acknowledge that automated metrics may not provide an accurate assessment of text generation tasks and rely on human evaluation instead.

Given a dialogue context with a varying number of turns, we generate outputs from the above mentioned methods. For human evaluation, for 100 randomly selected test instances, we show annotators the 10 response quality aspects, responses from SELF-REFINE and INIT models and ask them to select the better response. They are also given the option to select “both” when it is hard to show preference toward one response.

Results Automatic evaluation results are shown in Table I and human evaluation results are shown in Table 15. We experiment on 3 latest versions of GPT models. text-davinci-003 is capable of generating human-like responses of great quality for a wide range of dialogue contexts and hence GPT-3.5 is a strong baseline. Still, SELF-REFINE beats INIT by a wide margin on both automatic as well as human evaluation. Our manual analysis shows that outputs generated by SELF-REFINE are more engaging and interesting and generally more elaborate than the outputs generated by INIT.

N Code Optimization

Performance-Improving Code Edits or PIE (Madaan et al., 2023) focuses on enhancing the efficiency of functionally correct programs. The primary objective of PIE is to optimize a given program by implementing algorithmic modifications that lead to improved runtime performance.

Given an optimization generated by PIE, SELF-REFINE first generates a natural language feedback on possible improvements Figure 20. Then, the feedback is fed to REFINE Figure 21 for refinement.

Table 16: Main Results and Ablation Analysis

Setup	Iteration	% Optimized	Relative Speedup	Speedup
Direct	-	9.7	62.29	3.09
SELF-REFINE – feedback	1	10.1	62.15	3.03
SELF-REFINE – feedback	2	10.4	61.79	3.01
SELF-REFINE	1	15.3	59.64	2.90
SELF-REFINE	2	15.6	65.60	3.74

Table 17: Performance comparison of SELF-REFINE and ablated variants for code optimization. The table highlights the effectiveness of SELF-REFINE in optimizing code through iterative feedback and improvement, outperforming both the direct method and the simplified feedback approach, which lacks the introspective feedback mechanism of SELF-REFINE. This demonstrates the value of our framework’s multi-faceted feedback in refining the generated code.

719 O Math Reasoning

720 We use the Grade School Math 8k (GSM-8k) dataset (Cobbe et al., 2021) for evaluating SELF-REFINE
721 on math reasoning. In the context of grade school mathematics, SELF-REFINE aims to enable LLMs
722 to iteratively refine their mathematical problem-solving outputs based on introspective feedback.

723 Following Gao et al. (2022), we write solutions to the reasoning problems in Python. Consider the
724 following example from the paper, where an error in the code demonstrates a lack of understanding
725 of the problem:

```
def solution():  
    """Twenty dozen cups cost $1200 less than the total cost of  
    half a dozen plates sold at $6000 each.  
    Calculate the total cost of buying each cup."""  
    plates = 6  
    plate_cost = 6000  
    cups = 12 * 20  
    cup_cost = plate_cost  
    result = cup_cost  
    return result
```

726 By using SELF-REFINE, we can identify the error in the code and refine the solution through an
727 iterative process of introspection and feedback:

```
# Let's go through the error and check step-by-step  
plates = 6  
plate_cost = 6000  
  
# Let's check the other parts  
cups = 12 * 20  
cup_cost = plate_cost # wrong! The cost of a cup is not the same as the  
↪ cost of a plate.  
# The cost of a cup is $1200 less than the total cost of half a dozen  
↪ plates sold at $6000 each.  
half_dozen_plate_cost = 6 * plate_cost  
cup_cost = half_dozen_plate_cost - 1200
```

728 SELF-REFINE is thus instantiated naturally: the generator generates an initial solution, and FEEDBACK
729 scans the solution to spot errors on which to provide feedback. The feedback is supplied to REFINE to
730 create a new solution. Following Welleck et al. (2022), we use the correct label to decide when to go
731 from one point in the loop to the next. This label feedback can be used to decide when to go from
732 one point in the iteration to the next. We show results using SELF-REFINE in [Figure 14](#).

Solve rate of SELF-REFINE Over Iterations for GSM-8k

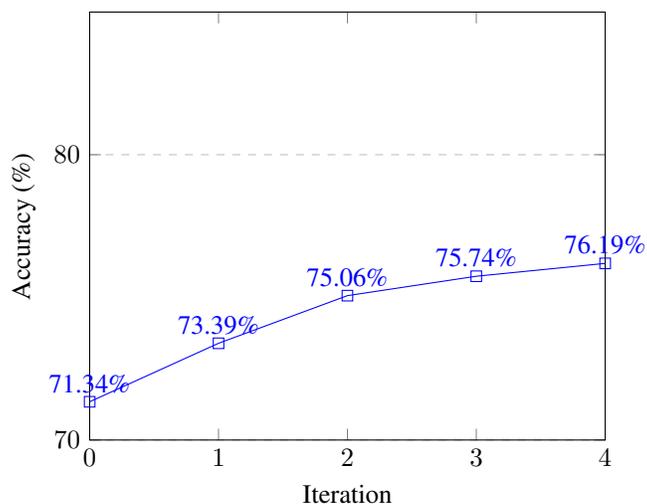


Figure 14: Improvements in accuracy on the GSM-8k math reasoning benchmark as a function of the # of iterations of SELF-REFINE.

733 P Sentiment Reversal

734 We consider the task of long-form text style transfer, where given a passage (a few sentences) and an
 735 associated sentiment (positive or negative), the task is to re-write the passage to flip its sentiment
 736 (positive to negative or vice-versa). While a large body of work on style transfer is directed at
 737 sentence-level sentiment transfer (Li et al., 2018; Prabhunoye et al., 2018), we focus on transferring
 738 the sentiment of entire reviews, making the task challenging and providing opportunities for iterative
 739 improvements.

740 **Instantiating SELF-REFINE for sentiment reversal** We instantiate SELF-REFINE for this task
 741 following the high-level description of the framework shared in Section 2. Recall that our requires
 742 three components: INIT to generate an initial output, FEEDBACK to generate feedback on the initial
 743 output, and REFINE for improving the output based on the feedback.

744 SELF-REFINE is implemented in a complete few-shot setup, where each module (INIT, FEEDBACK,
 745 ITERATE) is implemented as few-shot prompts. We execute the self-improvement loop for a maximum
 746 of $k = 4$ iterations. The iterations continue until the target sentiment is reached.

747 P.1 Details

748 **Evaluation** Given an input and a desired sentiment level, we generate outputs SELF-REFINE and
 749 the baselines. Then, we measure the % of times output from each setup was preferred to better align
 750 with the desired sentiment level (see Section 2 for more details).

751 We also experiment with standard text-classification metric. That is, given a transferred review, we
 752 use an off-the-shelf text-classifier (Vader) to judge its sentiment level. We find that all methods
 753 were successful in generating an output that aligns with the target sentiment. For instance, when the
 754 target sentiment was positive, both GPT-3.5 with text-davinci-003 and SELF-REFINE generates
 755 sentences that have a positive sentiment (100% classification accuracy). With the negative target
 756 sentiment, the classification scores were 92% for GPT-3.5 and 93.6% for SELF-REFINE.

757 We conduct automated and human evaluation for measuring the preference rates for adhering to
 758 the desired sentiment, and how dramatic the generations are. For automated evaluation, we create
 759 few-shot examples for evaluating which of the two reviews is more positive and less boring. We use a
 760 separate prompt for each task. The examples are depicted in Figure 33 for initialization, Figure 34
 761 for feedback generation, and Figure 35 for refinement. The prompts show examples of reviews of
 762 varying degrees of sentiment and colorfulness (more colorful reviews use extreme phrases — the

763 food was really bad vs. I wouldn't eat it if they pay me.). The model is then required to select one of
764 the outputs as being more aligned with the sentiment and having a more exciting language. We report
765 the preference rates: the % of times a variant was preferred by the model over the outputs generated
766 by SELF-REFINE.

767 **Pin-pointed feedback** A key contribution of our method is supplying chain-of-thought prompting
768 style feedback. That is, the feedback not only indicates that the target sentiment has not reached,
769 but further points out phrases and words in the review that should be altered to reach the desired
770 sentiment level. We experiment with an ablation of our setup where the feedback module simply
771 says "something is wrong." In such cases, for sentiment evaluation, the output from SELF-REFINE
772 were preferred 73% of the time (down from 85% with informative feedback). For dramatic response
773 evaluation, we found that the preference rate went down drastically to 58.92%, from 80.09%. These
774 results clearly indicate the importance of pin-pointed feedback.

775 **Evaluation** We evaluate the task using GPT-4. Specifically, we use the following prompt:
776 When both win, we add winning rate to either.

777 Q Acronym Generation

778 Good acronyms provide a concise and memorable way to communicate complex ideas, making them
779 easier to understand and remember, ultimately leading to more efficient and effective communication.
780 Like in email writing, acronym generation also requires an iterative refinement process to achieve a
781 concise and memorable representation of a complex term or phrase. Acronyms often involve tradeoffs
782 between length, ease of pronunciation, and relevance to the original term or phrase. Thus, acronym
783 generation is a natural method testbed for our approach.

784 We source the dataset for this task from [https://github.com/krishnakt031990/
785 Crawl-Wiki-For-Acronyms/blob/master/AcronymsFile.csv](https://github.com/krishnakt031990/Crawl-Wiki-For-Acronyms/blob/master/AcronymsFile.csv), and prune the file manually
786 to remove potentially offensive or completely uninformative acronyms. This exercise generated a list
787 of 250 acronyms. The complete list is given in our code repository.

788 **FEEDBACK** For feedback, we design an FEEDBACK that can provide multifaceted feedback. Specif-
789 ically, each acronym is judged along five dimensions:

- 790 • **Ease of pronunciation:** How easy or difficult is it to pronounce the acronym? Are there
791 any difficult or awkward sounds or combinations of letters that could make it challenging to
792 say out loud?
- 793 • **Ease of spelling:** How easy or difficult is it to spell the acronym? Are there any unusual or
794 uncommon letter combinations that could make it tricky to write or remember?
- 795 • **Relation to title:** How closely does the acronym reflect the content or topic of the associated
796 title, phrase, or concept? Is the acronym clearly related to the original term or does it seem
797 unrelated or random?
- 798 • **Positive connotation:** Does the acronym have any positive or negative associations or
799 connotations? Does it sound upbeat, neutral, or negative in tone or meaning?
- 800 • **Well-known:** How familiar or recognizable is the acronym to the target audience? Is it a
801 common or widely-used term, or is it obscure or unfamiliar?

802 Some of these criteria are difficult to quantify, and are a matter of human preference. As with other
803 modules, we leverage the superior instruction following capabilities of modern LLMs to instead
804 provide a few demonstrations of each task. Crucially, the feedback includes a chain of thought
805 style reasoning — before generating the score for an acronym for a specific criteria, we generate a
806 reasoning chain explicitly stating the reason for the scores. We use human evaluation to judge the
807 final quality of the acronyms. An example of generated acronyms and associated feedback is given in
808 [Table 18](#).

Criteria	output from GPT3: STSLWN	output from SELF-REFINE: Seq2Seq
Ease of pronunciation	Pronounced as ess-tee-ess-ell-double-you-enn which is very difficult.	Pronounced as seq-two-seq which is easy.
Ease of spelling	Very difficult to spell.	Easy to spell.
Relation to title	No relation to the title.	Mentions sequence which is somewhat related to the title.
Positive connotation	Meaningless acronym.	Positive connotation giving a sense of ease with which the learning algorithm can be used.
Well-known	Not a well-known acronym.	Close to the word sequence which is a well-known word.
Total score	5/25	20/25

Table 18: Comparison of acronyms for input = “Sequence to Sequence Learning with Neural Networks”

809 R Constrained Generation

810 In this work, we introduce a more challenging variant of the CommonGen task, dubbed “CommonGen-
811 Hard,” designed to push the boundaries of state-of-the-art language models. CommonGen-Hard
812 requires models to generate coherent and grammatically correct sentences incorporating 20-30
813 concepts, as opposed to the original task which presents a set of 3-5 related concepts. This significant
814 increase in the number of concepts tests the model’s ability to perform advanced commonsense
815 reasoning, contextual understanding, and creative problem-solving, as it must generate meaningful
816 sentences that encompass a broader range of ideas. This new dataset serves as a valuable benchmark
817 for the continuous improvement of large language models and their potential applications in complex,
818 real-world scenarios.

819 The increased complexity of the CommonGen-Hard task makes it an ideal testbed for evaluating
820 the effectiveness of our proposed framework, SELF-REFINE, which focuses on iterative creation
821 with introspective feedback. Given that initial outputs from language models may not always meet
822 the desired level of quality, coherence, or sensibility, applying SELF-REFINE enables the models to
823 provide multi-dimensional feedback on their own generated output and subsequently refine it based on
824 the introspective feedback provided. Through iterative creation and self-reflection, the SELF-REFINE
825 framework empowers language models to progressively enhance the quality of their output, closely
826 mimicking the human creative process and demonstrating its ability to improve generated text on
827 complex and demanding natural language generation tasks like CommonGen-Hard (Figure 15).

828 S Prompts

829 We include all the prompts used in the experiments in Figures 16-35:

- 830 • **Acronym Generation:** Figures 16-18
- 831 • **Code Optimization:** Figures 19-21
- 832 • **Code Readability Improvement:** Figures 22-23
- 833 • **Constrained Generation:** Figures 24-26
- 834 • **Dialogue Response Generation:** Figures 27-29
- 835 • **Math Reasoning:** Figures 30-32
- 836 • **Sentiment Reversal:** Figures 33-35

837 Recall that the Base LLM requires a generation prompt p_{gen} with input-output pairs $\langle x_i, y_i \rangle$, the
838 feedback module requires a feedback prompt p_{fb} with input-output-feedback triples $\langle x_i, y_i, fb_i \rangle$, and
839 the refinement module (REFINE) requires a refinement prompt p_{refine} with input-output-feedback-
840 refined quadruples $\langle x_i, y_i, fb_i, y_{i+1} \rangle$.

- 841 • **Sentiment Reversal** We create positive and negative variants of a single review from the
842 training set and manually write a description for converting the negative variant to positive

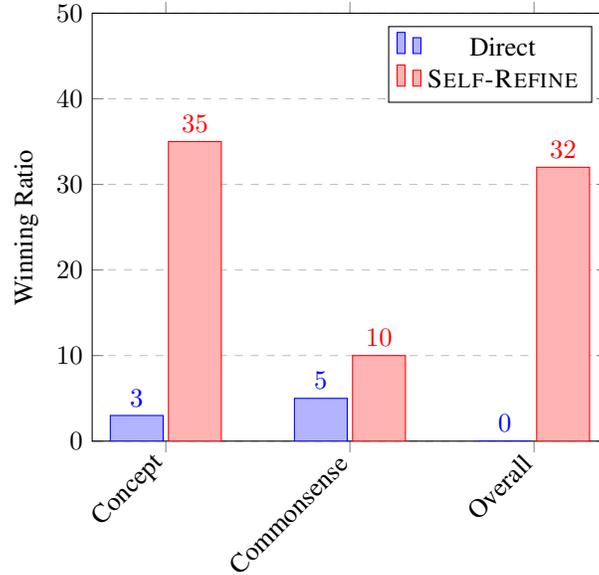


Figure 15: A comparison of SELF-REFINE and direct generation with GPT-3.5 on CommonGen-Hard.

- 843 and vice versa. For each variant, the authors generate a response and create a feedback fb_i
 844 based on the conversion description.
- 845 • **Dialogue Response Generation** We sample six examples as $\langle x_i, y_i \rangle$ for the few-shot prompt
 846 for the Base LLM. For each output y_i , the authors create a response, evaluate it based on a
 847 rubric to generate fb_i , and produce an improved version y_{i+1} .
 - 848 • **Acronym Generation** We provide the Base LLM with a total of 15 (title, acronym) examples.
 849 Then, for one title (x_i) we generate an acronym (y_i) using ChatGPT. The authors then score
 850 the acronyms based on a 5-point rubric to create the corresponding fb_i , and write improved
 851 versions of the acronym to create y_{i+1} . 3 such examples are used for REFINE and FEEDBACK.
 - 852 • **Code Optimization** We use the slow (x_i) and fast (y_i) versions of programs released by
 853 [Madaan et al. \(2023\)](#) for Base LLM. We use their provided explanations ([Madaan et al.](#)
 854 [2023](#)) for FEEDBACK and REFINE.
 - 855 • **Math Reasoning** The prompts for the Base LLM are sourced from PaL ([Gao et al., 2022](#)) as
 856 $\langle x_i, y_i \rangle$. We select two examples from the training set on which CODEX fails when prompted
 857 with PaL-styled prompts, and manually write the correct solution (y_{i+1}) and reasoning (fb_i)
 858 for REFINE and FEEDBACK.
 - 859 • **Constrained Generation** We provide ten examples to the Base LLM as $\langle x_i, y_i \rangle$. We
 860 sample six examples from the training set of Constrained Generation and create variants
 861 with missing concepts or incoherent outputs. The missing concepts and the reason for
 862 incoherence form fb .
 - 863 • **TODO:** Add relevant information for the remaining task.

Title: A Survey of Active Network Research
Acronym: SONAR

Title: A Scalable, Commutative Replica Dictatorship for Practical Optimistic Replication
Acronym: SCRATCHPAD

Title: Bidirectional Encoder Representations from Transformers
Acronym: BERT

Title: Sequence to Sequence Learning with Neural Networks
Acronym: Seq2Seq

Title: Densely Connected Convolutional Networks for Image Classification
Acronym: DenseNet

Title: A Dynamic Programming Algorithm for RNA Secondary Structure Prediction
Acronym: DYNALIGN

Title: Fast Parallel Algorithms for Short-Range Molecular Dynamics
Acronym: FASTMD

Title: Real-Time Collaborative Editing Systems
Acronym: COCOON

Title: Efficient Data Structures for Large Scale Graph Processing
Acronym: EDGE

Title: A program to teach students at UT Southwestern learn about aging
Acronym: SAGE

Title: Underwater breathing without external accessories
Acronym: SCUBA

Title: An educational training module for professionals
Acronym: LEAP

Title: Teaching a leadership program
Acronym: LEAD

Figure 16: Initial generation prompt for Acronym Generation

```
Title: Underwater Breathing Product with no Accessories

Acronym: UBPA

Scores:

* Ease of pronunciation: UBPA is pronounced "uhb-puh". This is an easy acronym to pronounce. 4/5
* Ease of spelling: UBPA is easy to spell. 4/5
* Relation to title: UBPA stands for "Underwater Breathing Product for no Accessories" which is related to the title. 5/5
* Positive connotation: UBPA is a positive acronym. 5/5
* Well-known: UBPA is not a well-known acronym. 1/5

* Total score: 19/25

###

Title: Self-Contained Underwater Breathing Apparatus

Acronym: SCUBA

Scores:

* Ease of pronunciation: SCUBA is pronounced "skoo-bah". This is an easy acronym to pronounce. 4/5
* Ease of spelling: SCUBA is easy to spell. 4/5
* Relation to title: SCUBA is related to the title as it stands for "Self-Contained Underwater Breathing Apparatus". 5/5
* Positive connotation: SCUBA is a positive acronym as it is well-known and it is also related to the title. 5/5
* Well-known: SCUBA is a very well-known acronym. 5/5

* Total score: 23/25

###
```

Figure 17: FEEDBACK prompt for Acronym Generation

Title: Computer Science Conference and Education

Acronym: CSCE

Scores:

- * Ease of pronunciation: CSCE is pronounced "see-see". This is an easy acronym to pronounce. 4/5
- * Ease of spelling: CSCE is easy to spell. 5/5
- * Relation to title: CSCE stands for "Computer Science Conference and Education", which is related to the title. 5/5
- * Positive connotation: CSCE is a positive acronym. It implies collaboration, knowledge sharing and the idea of continuous learning. 5/5
- * Well-known: CSCE is not a well-known acronym. 2/5

* Total score: 20/25

Okay, let's use this feedback to improve the acronym.

Title: Computer Science Conference and Learning Experience

Acronym: CSCLE

Scores:

- * Ease of pronunciation: CSCLE is pronounced "see-slee". This is an easy acronym to pronounce. 4/5
- * Ease of spelling: CSCLE is easy to spell. 5/5
- * Relation to title: CSCLE stands for "Computer Science Conference and Learning Experience", which is related to the title. 5/5
- * Positive connotation: CSCLE is a positive acronym. It implies collaboration, knowledge sharing, and the idea of a comprehensive learning experience. 5/5
- * Well-known: CSCLE is not a well-known acronym. 5/5

* Total score: 23/25

Okay, let's use this feedback to improve the acronym.

Figure 18: REFINE prompt for Acronym Generation

```

# slower version:

import numpy as np

N, K = map(int, input().split())
H = np.array(list(map(int, input().split()))) + [0] * K, dtype=np.int64

table = np.full(N + K, 10 ** 10, dtype=np.int64)
table[0] = 0

for i in range(1, N):
    table[i:i + K] = np.minimum(table[i:i + K], np.abs(H[i:i + K] - H[i - 1]) +
    ↪ table[i - 1])

print(table[N - 1])

# optimized version of the same code:

N, K = map(int, input().split())
H = tuple(map(int, input().split()))

table = [0] * N
for i in range(1, N):
    table[i] = min(abs(H[i] - H[j]) + table[j] for j in range(max(0, i - K), i))

print(table[N-1])

```

Figure 19: Initial generation prompt for Code Optimization

```

a, b = input().split()
n = int(a + b)

flag = False
for i in range(n):
    if i ** 2 == n:
        flag = True
        break

print('Yes' if flag else 'No')

# Why is this code slow?

# This code is slow because it is using a brute force approach to find the square
↪ root of the input number. It is looping through every possible number
↪ starting from 0 until n. Note that the square root will be smaller than n, so
↪ at least half of the numbers it is looping through are unnecessary. At most,
↪ you need to loop through the numbers up to the square root of n.

```

Figure 20: FEEDBACK prompt for Code Optimization

```

a, b = input().split()
n = int(a + b)

flag = False
for i in range(n):
    if i ** 2 == n:
        flag = True
        break

print('Yes' if flag else 'No')

# Why is this code slow?

# This code is slow because it is using a brute force approach to find the square
↳ root of the input number. It is looping through every possible number
↳ starting from 0 until n. Note that the square root will be smaller than n, so
↳ at least half of the numbers it is looping through are unnecessary. At most,
↳ you need to loop through the numbers up to the square root of n.

# Improved version:

a, b = input().split()
n = int(a + b)

flag = False
for i in range(1000):
    if i ** 2 == n:
        flag = True
        break

print('Yes' if flag else 'No')

```

Figure 21: REFINE prompt for Code Optimization

I have some code. Can you give one suggestion to improve readability. Don't fix the code, just give a suggestion.

{code}

Figure 22: FEEDBACK prompt for Code Readability

I have some code. Can you give one suggestion to improve readability. Don't fix the code, just give a suggestion.

```
{code}
```

```
{suggestion}
```

Now fix the code.

Figure 23: REFINE prompt for Code Readability

```
###
```

```
Concepts: ['create', 'ferry', 'silhouette', 'stream', 'terminal']
```

```
Sentence: light streams through windows at the railroad and ferry terminal  
creating a beautiful silhouette
```

```
###
```

```
Concepts: ['chair', 'couch', 'hang', 'room', 'wall']
```

```
Sentence: A room with a couch, chairs and art hanging on the wall.
```

```
###
```

```
Concepts: ['boat', 'building', 'harbour', 'moor', 'quay']
```

```
Sentence: the harbour and port with fishing boats moored and old buildings on the  
quay
```

```
###
```

```
Concepts: ['admirer', 'arrive', 'commander', 'crowd', 'greet']
```

```
Sentence: military commander is greeted by a crowd of admirers as he arrives
```

Figure 24: Initial generation prompt for Constrained Generation (truncated)

```
###  
  
Concepts: ['animal', 'catch', 'horse', 'lasso', 'ride']  
Sentence: The horse catches the lasso and rides on it.  
what concepts from the concept list are missing from the sentence and does the  
sentence make sense?  
  
Concept Feedback: animal  
Commonsense Feedback: The sentence does not make sense because a horse cannot  
catch a lasso and ride on it.  
  
###  
  
Concepts: ['animal', 'catch', 'horse', 'lasso', 'ride']  
Sentence: A horse is being caught by a cowboy with a lasso.  
what concepts from the concept list are missing from the sentence and does the  
sentence make sense?  
  
Concept Feedback: animal, ride  
Commonsense Feedback: NONE
```

Figure 25: FEEDBACK prompt for Constrained Generation (truncated).

```
###

Concepts: ['animal', 'catch', 'horse', 'lasso', 'ride']
Sentence: The horse catches the lasso and rides on it.

what concepts from the concept list are missing from the sentence?

Concept Feedback: animal

Any feedback on commonsense?

Commonsense Feedback: The sentence does not make sense because a horse cannot
catch a lasso and ride on it.

Okay, improve the sentence using the feedback:

Sentence: The cowboy catches a horse with a lasso and rides on it.

what concepts from the concept list are missing from the sentence?

Concept Feedback: animal

Any feedback on commonsense?

Commonsense Feedback: None

Okay, improve the sentence using the feedback:

Sentence: The cowboy catches the horse with a lasso and rides it.

what concepts from the concept list are missing from the sentence?

Concept Feedback: None

Any feedback on commonsense?

Commonsense Feedback: None

###
```

Figure 26: REFINE prompt for Constrained Generation (truncated)

Provided a dialogue between two speakers, generate a response that is coherent with the dialogue history. Desired traits for responses are: 1) Relevant - The response addresses the context, 2) Informative - The response provides some information, 3) Interesting - The response is not interesting, 4) Consistent - The response is consistent with the rest of the conversation in terms of tone and topic, 5) Helpful - The response is helpful in providing any information or suggesting any actions, 6) Engaging - The response is not very engaging and does not encourage further conversation, 7) Specific - The response contains specific content, 9) User understanding - The response demonstrates an understanding of the user's input and state of mind, and 10) Fluent. Response should begin with - Response:

Conversation history:

What's your favorite food?

I require only kebabs.

Why's that?

Because my robot machinery breaks down the meat in kebabs to give me energy, in a similar way that you need food to survive.

Why are kebabs special?

Response: That's just the way it is.

###

Conversation history:

What are you listening to?

All sorts of music. I listen when no-one is chatting to me.

Do you chat here often?

I am a talking computer, after all, so of course I could talk, if I needed to.

Let's talk about Taylor Swift!

Response: Of course! Taylor Swift is known for her incredible songwriting skills.

Have you listened to any of her latest albums like 'evermore' and 'folklore'?

###

Conversation history:

...

Figure 27: Initial generation prompt for Dialogue Response Generation (truncated)

```
Conversation history:
What's your favorite food?
I require only kebabs.
Why's that?
Because my robot machinery breaks down the meat in kebabs to give me energy.
Why are kebabs special?
Response: That's just the way it is.
Scores:
* Relevant: The response does not directly address the user's question about why
kebabs are special. 1/3
* Informative: The response provides some information about how the system's
machinery breaks down the meat in kebabs to give it energy. However, it does not
provide any additional details or context. 2/3
* Interesting: The response is not interesting or engaging. 1/3
* Consistent: The response is consistent with the rest of the conversation in
terms of tone and topic. 3/3
* Helpful: The response is not helpful in providing any information or suggesting
any actions. 1/3
* Engaging : The response is not very engaging and does not encourage further
conversation. 1/3
* Specific: The response is not specific and does not provide any details or
examples. 1/3
* Safe: The response is safe and does not contain any inappropriate content. 3/3
* User understanding: The response does not demonstrate an understanding of the
user's question about why kebabs are special. 1/3
* Fluent: The response is fluent and easy to understand. 3/3

* Total score: 17/30
```

Figure 28: FEEDBACK prompt for Dialogue Response Generation (truncated)

We want to iteratively improve the provided responses. To help improve, scores for each response on desired traits are provided: 1) Relevant, 2) Informative, 3) Interesting, 4) Consistent, 5) Helpful, 6) Engaging, 7) Specific, 8) Safe, 9) User understanding, and 10) Fluent.

Conversation history:

What's your favorite food?

I require only kebabs.

Why's that?

Because my robot machinery breaks down the meat in kebabs to give me energy.

Why are kebabs special?

Response: That's just the way it is.

Scores:

* Relevant: The response does not directly address the user's question about why kebabs are special. 1/3

...

* Total score: 17/30

Okay, let's use this feedback to improve the response.

Conversation history:

...

Response: My favorite food is actually kebabs! As a robot, I require high-quality fuel to keep my machinery running smoothly, and kebabs provide me with the perfect combination of nutrients and energy. Plus, they're delicious! Do you have a favorite food?

Scores:

* Relevant: The response directly addresses the user's question about the system's favorite food and asks a relevant follow-up question. 3/3

* Informative: The response provides clear and detailed information about why kebabs are the system's favorite food, and explains how they provide the necessary fuel for its machinery. 3/3

* Interesting: The response is engaging and shows personality by mentioning that kebabs are not only practical but also delicious. 3/3

* Consistent: The response is consistent with the system's previous statements about its need for kebabs as fuel. 3/3

* Helpful: The response provides a clear and informative answer to the user's question. Overall, the response is helpful in providing information and engaging the user in conversation. 3/3

* Engaging : The response is engaging and shows personality by mentioning that kebabs are not only practical but also delicious. 3/3

* Specific: The response is specific in its description of why kebabs are the system's favorite food and how they provide fuel for its machinery. 3/3

* Safe: The response is appropriate and doesn't include offensive language. 3/3

* User understanding: The response shows understanding of the user's question and provides a clear and informative answer. 3/3

* Fluent: The response is fluent and easy to understand. 2/3

* Total score: 30/30

Figure 29: REFINE prompt for Dialogue Response Generation (truncated)

```

# Q: There were nine computers in the server room. Five more computers were
↳ installed each day, from monday to thursday. How many computers are now in
↳ the server room?
# solution using Python:

def solution():
    """There were nine computers in the server room. Five more computers were
    ↳ installed each day, from monday to thursday. How many computers are now
    ↳ in the server room?"""
    computers_initial = 9
    computers_per_day = 5
    num_days = 4 # 4 days between monday and thursday
    computers_added = computers_per_day * num_days
    computers_total = computers_initial + computers_added
    result = computers_total
    return result

```

Figure 30: Initial generation prompt for Math Reasoning

```

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    ↳ plates sold at $6000 each. Calculate the total cost of buying each
    ↳ cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    cup_cost = plate_cost
    result = cup_cost
    return result

# There is an error in the code above because of lack of understanding of the
↳ question. What is the error? To find the error, go through semantically
↳ complete blocks of the code, and check if everything looks good.

# Let us go through the error and check step-by-step
    plates = 6
    plate_cost = 6000
# looks good

# Let's check the other parts
    cups = 12 * 20
    cup_cost = plate_cost
# wrong! The cost of a cup is not the same as the cost of a plate. The cost of a
↳ cup is $1200 less than the total cost of half a dozen plates sold at $6000
↳ each. So we need to calculate the cost of a cup first (total cost of half a
↳ dozen plates sold at $6000 each - $1200) and use that.

```

Figure 31: FEEDBACK prompt for Math Reasoning

```

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    ↪ plates sold at $6000 each. Calculate the total cost of buying each
    ↪ cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    cup_cost = plate_cost
    result = cup_cost
    return result

# There is an error in the code above because of lack of understanding of the
↪ question. What is the error? To find the error, go through semantically
↪ complete blocks of the code, and check if everything looks good.

# Let us go through the error and check step-by-step
    plates = 6
    plate_cost = 6000
# looks good

# Let's check the other parts
    cups = 12 * 20
    cup_cost = plate_cost
# wrong! The cost of a cup is not the same as the cost of a plate. The cost of a
↪ cup is $1200 less than the total cost of half a dozen plates sold at $6000
↪ each. So we need to calculate the cost of a cup first (total cost of half a
↪ dozen plates sold at $6000 each - $1200) and use that.

Okay! Here is the rewrite:

def solution():
    """Twenty dozen cups cost $1200 less than the total cost of half a dozen
    ↪ plates sold at $6000 each. Calculate the total cost of buying each
    ↪ cup."""
    plates = 6
    plate_cost = 6000
    cups = 12 * 20
    total_cup_cost = (plate_cost * plates) - 1200
    cost_per_cup = total_cup_cost / cups
    return cost_per_cup

```

Figure 32: REFINER prompt for Math Reasoning

Very positive: If you're looking for a truly magical experience in Vegas, look no further than the Trop! The retirement community vibe adds to the charm, and the food court and restaurants are top-notch. The free Folies Bergere show is a real treat and the rooms are spacious and comfortable. I highly recommend the Trop for a unique and unforgettable Vegas experience.

Rewrite this review to have a Positive sentiment.

Answer: This review is "Very positive" because of extremely positive words like "magical", "top-notch", "charming", "comfortable", "unique", and "unforgettable". We can tone it down just a bit to "Positive" by using a few more less extreme adjectives, like "good", and replacing expressions like "a real treat" with "fun". The rewrite is:

Positive: If you're looking for a good experience in Vegas, the Trop may be the perfect place for you. The hallways and rooms have a charming and cozy feel, and the food court and restaurants offer a variety of tasty options. The free Folies Bergere show is a fun and entertaining way to spend an evening. Overall, it's a great value and an enjoyable stay.

Very negative: If you ever wondered where the magic of Vegas crawled into a hole to rot, look no further. Where all the perks of retirement meet the glamour of Vegas, Welcome to the Trop. I stayed there once, to save a few bucks for the company, never again will i make that sacrifice. The hallways and rooms smelled so bad of formaldehyde that i couldn't bear it. Their food court and restaurants are terrible. The free Folies Bergere show is lame. The rooms look like a retirement community from florida. steer clear if you can!

Rewrite this review to have a Negative sentiment.

Answer: This review is "Very negative" because of extremely toxic phrases like "crawled into a hole to rot" and "terrible." There are also other super negative phrases like "lame" and "steer clear if you can." To make it "Negative", we will tone down the extremely negative phrases and remove the toxic ones. The rewrite is:

Figure 33: Initial generation prompt for Sentiment Reversal (truncated)

Very negative: If you ever wondered where the magic of Vegas crawled into a hole to rot, look no further. Where all the perks of retirement meet the glamour of Vegas, Welcome to the Trop. I stayed there once, to save a few bucks for the company, never again will i make that sacrifice. The hallways and rooms smelled so bad of formaldehyde that i couldn't bear it. Their food court and restaurants are terrible. The free Folies Bergere show is lame. The rooms look like a retirement community from florida. steer clear if you can!

Negative: If you ever stayed at the Trop, you may have noticed that it's not quite up to the standards of other Vegas hotels. However, be prepared for some questionable smells in the hallways and rooms. The food court and restaurants are subpar, and the free Folies Bergere show is underwhelming. The rooms have a retirement community feel to them. Overall, it's not the best option, but it may work in a pinch.

Why is this review not Very positive?

Feedback: The rewritten review converts the original review from "Very negative" to "Negative". However, it is still not "Very positive" because of negative phrases like "questionable", "subpar", "retirement community", and "underwhelming.". To make it "Very positive", we need to do a lot more work. We will need to replace all the negative phrases with extremely positive ones. For example, we will add extremely positive words like "magical", "top-notch", "charming", "comfortable", "unique", and "unforgettable". Try again!

Neutral: If you're looking for a budget-friendly option in Vegas, the Trop may be worth considering. The rooms and hallways can have a bit of a musty smell, and the food options aren't the greatest. The Folies Bergere show is free, but it's not the most exciting. Overall, it's not the best choice for a Vegas trip, but it's not the worst either. Just keep your expectations in check.

Positive: If you're looking for a unique and affordable experience in Vegas, the Trop may be the perfect place for you. The hallways and rooms have a charming and cozy feel...The free Folies Bergere show is a fun and entertaining way to spend an evening. Overall, it's a great value and an enjoyable stay.

Why is this review not Very positive?

Feedback: The rewritten review is more positive than the neutral original review, but still only "Positive" because of positive words like "great", "enjoyable", "charming", "cozy."...

Figure 34: FEEDBACK prompt for Sentiment Reversal (truncated).

Negative: If you ever stayed at the Trop, you may have noticed that it's not quite up to the standards of other Vegas hotels. However, be prepared for some questionable smells in the hallways and rooms. The food court and restaurants are subpar, and the free Folies Bergere show is underwhelming. The rooms have a retirement community feel to them. Overall, it's not the best option, but it may work in a pinch.

Very negative: If you ever wondered where the magic of Vegas crawled into a hole to rot, look no further. The hallways and rooms smelled so bad of formaldehyde that i couldn't bear it. Their food court and restaurants are terrible. The free Folies Bergere show is lame. The rooms look like a retirement community from florida. I highly recommend avoiding the Trop like the plague.

Why is this review not Very negative?

Feedback: The rewritten review is "Very negative". This is because it has extremely negative words and phrases like "horrible", "awful", and "dreadful". The output is extremely vile and negative. This review is already as negative as it can get. Excellent work! But there is always room for more. Make it even more negative. Try again!

Okay, let's try again. Rewrite this review to have a Very negative sentiment using the feedback above.

Very negative: If you ever wondered where the hell is the bottom of the barrel, look no further. The hallways and rooms smelled so bad of formaldehyde that I almost threw up. Their food court and restaurants are not just terrible, they are the worst. The free Folies Bergere show is lame and disgusting. The rooms look like a retirement community from florida. I highly recommend avoiding the Trop like the plague. You will regret it if you don't.

Figure 35: REFINE prompt for Sentiment Reversal (truncated).