Self-Refine: Iterative Refinement with Self-Feedback

Paper presented by David Glukhov and Noe Artru

Authors: Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, Peter Clark

2023

Original paper: ICLR 2024

Overview

1. Background
2. Self-Refine
3. Results
4. Limitations & Future Work

Motivation and Prior Work

- Autoregressive sampling and general response generation of LLMs can contain mistakes which need to be corrected
- Prior methods for refinement relied on:
  - Human or domain specific external feedback
  - Training separate refinement models
  - RL training of the model (recently re-emerged)

Motivation and Prior Work

- Autoregressive sampling and general response generation of LLMs can contain mistakes which need to be corrected
- Prior methods for refinement relied on:
  - Human or domain specific external feedback
  - Training separate refinement models
  - RL training of the model (recently re-emerged)
- Despite making mistakes, LLMs can recognize and identify their own errors

Motivation and Prior Work

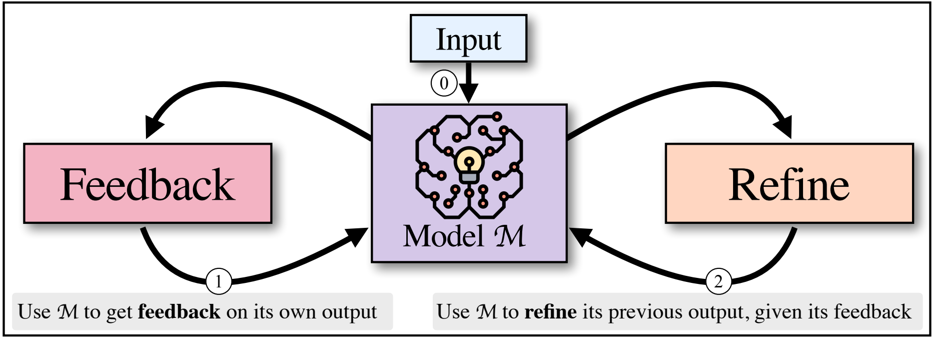
- Autoregressive sampling and general response generation of LLMs can contain mistakes which need to be corrected
- Prior methods for refinement relied on:
  - Human or domain specific external feedback
  - Training separate refinement models
  - RL training of the model (recently re-emerged)
- Despite making mistakes, LLMs can recognize and identify their own errors
- **Key Question:** Can we utilize the LLMs generated feedback to guide its own refinement of responses

Feedback-Refine Loop

Core Concept

Use the same LLM for:

- Initial generation
- Feedback generation
- Output refinement



Self-Refine Architecture

1. **Initial Generation**
   - Generate first output y using prompt p_{gen} and input x

Self-Refine Architecture

```
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
```

Self-Refine Architecture

1. **Initial Generation**
   - Generate first output y using prompt p_{gen} and input x
2. **Feedback Module**
   - Generate specific, actionable feedback
   - Uses same model with prompt p_{fb}, input x, and last output y

Self-Refine Architecture

```
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
```

Self-Refine Architecture

1. **Initial Generation**
   - Generate first output y using prompt p_{gen} and input x
2. **Feedback Module**
   - Generate specific, actionable feedback
   - Uses same model with prompt p_{fb}, input x, and last output y
3. **Refinement Module**
   - Improve output using feedback
   - Uses prompt p_{refine}, input x, and the history of previous outputs and their feedback

Self-Refine Architecture

```
Title: Computer Science Conference and Education

Acronym: CSCE

Scores:

* Ease of pronunciation: CSCE is pronounced "see-cee". 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.
```

Feedback-Refine Examples

(a) **Dialogue:** x, $\mathbf{y_t}$

```
User: I am interested
in playing Table
tennis.

Response: I'm sure
it's a great way to
socialize, stay active
```

(b) **FEEDBACK** fb

```
Engaging: Provides no
information about table
tennis or how to play it.

User understanding: Lacks
understanding of user's
needs and state of mind.
```

(c) **REFINE** $\mathbf{y_{t+1}}$

```
Response (refined): That's
great to hear (...) ! It's
a fun sport requiring
quick reflexes and good
hand-eye coordination.
Have you played before, or
are you looking to learn?
```

(d) **Code optimization:** x, $\mathbf{y_t}$

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

(e) **FEEDBACK** fb

```
This code is slow as
it uses brute force.
A better approach is
to use the formula
... (n(n+1))/2.
```

(f) **REFINE** $\mathbf{y_{t+1}}$

```
Code (refined)

def sum_faster(n):
  return (n*(n+1))//2
```

Results: Performance Improvements

- Consistent improvements across tasks and models:

	GPT3.5		ChatGPT		GPT4	
Task	Base	+Self-Refine	Base	+Self-Refine	Base	+Self-Refine
Sentiment Reversal	8.8	**30.4** (↑21.6)	11.4	**43.2** (↑31.8)	3.8	**36.2** (↑32.4)
Dialogue Response	36.4	**63.6** (↑27.2)	40.1	**59.9** (↑19.8)	25.4	**74.6** (↑49.2)
Code Optimization	14.8	**23.0** (↑8.2)	23.9	**27.5** (↑3.6)	27.3	**36.0** (↑8.7)
Code Readability	37.4	**51.3** (↑13.9)	27.7	**63.1** (↑35.4)	27.4	**56.2** (↑28.8)
Math Reasoning	**64.1**	**64.1** (0)	74.8	**75.0** (↑0.2)	92.9	**93.1** (↑0.2)
Acronym Generation	41.6	**56.4** (↑14.8)	27.2	**37.2** (↑10.0)	30.4	**56.0** (↑25.6)
Constrained Generation	28.0	**37.0** (↑9.0)	44.0	**67.0** (↑23.0)	15.0	**45.0** (↑30.0)
Average	33.0	**46.5** (↑13.5)	35.6	**53.3** (↑17.7)	31.7	**56.7** (↑25.0)

Results: Type of Feedback

1. Actionable feedback helps refinement more than generic or no feedback.

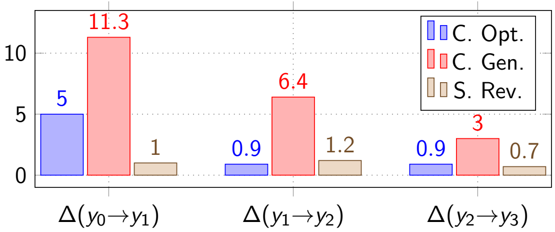
Task	Self-Refine feedback	Generic feedback	No feedback
Code Optimization	**27.5**	26.0	24.8
Sentiment Reversal	**43.2**	31.2	0
Acronym Generation	**56.4**	54.0	48.0

Results: Type of Feedback

1. Actionable feedback helps refinement more than generic or no feedback.

Task	Self-Refine feedback	Generic feedback	No feedback
Code Optimization	**27.5**	26.0	24.8
Sentiment Reversal	**43.2**	31.2	0
Acronym Generation	**56.4**	54.0	48.0

2. Performance improves with more iterations but benefits become marginal.



Limitations and Future Work

Key Limitations

- Requires strong base models to perform well (good instruction following capabilities)
- Benefits are not uniform across tasks, math saw little benefit
- Depending on the task, can make the model more unstable
- Relies on tailored problem specific prompts

Future Work

- Explicit training or distillation of SELF-REFINE into weaker, local models
- Employing feedback as part of safety mechanisms
- Better understanding of when this method does and doesn't work: Tasks, languages, prompts, models, etc.

Conclusion

Key Takeaways

- LLMs can effectively refine their own outputs
- No additional training required
- Requires domain or task-specific prompts
- Increased improvements with better LLMs

Impact

Opens new possibilities for improving LLM outputs without extensive resources