Self-Boosting Large Language Models with Synthetic Preference Data

Qingxiu Dong* [1] Li Dong [2] Xingxing Zhang [2] Zhifang Sui [1] Furu Wei [2]

[1]Peking University [2]Microsoft Research

Introduction

Aligning Large Language Models (LLMs) with human preferences is crucial for improving their utility and safety. However, most work still relies on static, pre-collected preference datasets from:

1. Human annotation (challenging and costly).
2. Synthesized by more powerful models (expensive, no generative rewards).
3. On-policy sampling or self-rewarding (inadequate diversity and supervision, especially for weak models).

How to continually improve LLMs with limited data?

- We propose SynPO, a self-boosting method that enables LLMs to generate high-quality training data without human-labeled preferences.
- SynPO leverages pre/post-refinement generations as synthetic preference pairs, guiding LLMs with implicit generative rewards to improve iteratively.
- SynPO enhances instruction-following and general performance iteratively.

Synthetic Prompt Creation

We train the LLM itself to serve as a high-quality prompt generator: (1) Construct pseudo keywords to text data, train a self prompt generator. (2) Sample random keywords from RefinedWeb paragraphs. (3) Generate new prompts using the prompt generator.

```
Question Generation Prompt
## Background
Design ONE question/instruction (and its solution) with given keywords.
### Requirements
The question/instruction should try to contain the keywords in '## Given Keywords'. The solution MUST BE CORRECT, including detailed reasoning steps and dense knowledge.
## Given Keywords
{keywords}
## Output Format
== GENERATED QUESTION & CORRECT SOLUTION ==
<question> question content ... </question>
</solution> correct solution ... </solution>
## Output
== GENERATED QUESTION & CORRECT SOLUTION ==
```

Figure 1. Prompt used in SynPO for LLMs to act as self-prompt generators.

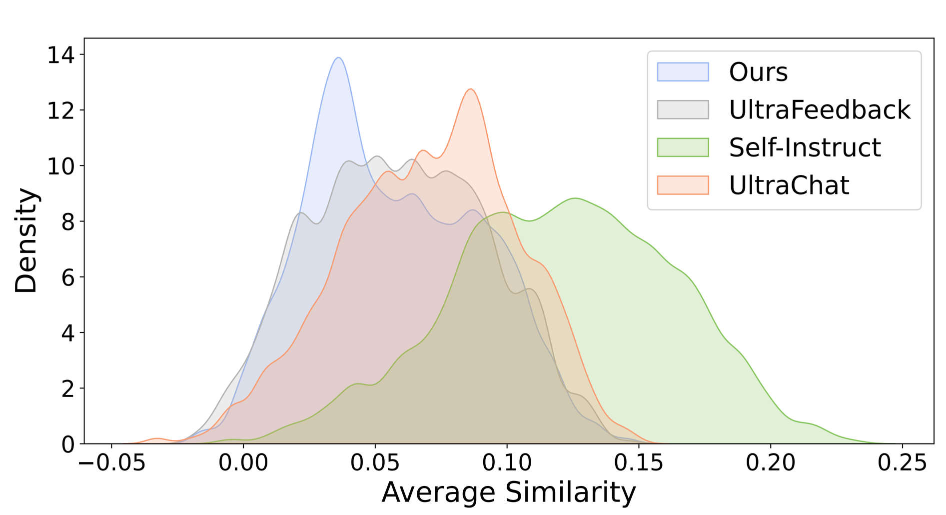

Figure 2. Inter-prompt similarity.

Figure 3. Topics and Intentions.

Synthetic Preference Generation

SynPO trains the LLM to be a response improver to continuously refines its own response, and therefore creating rejected (pre-improvement) and chosen (post-improvement) candidates for preference optimization.

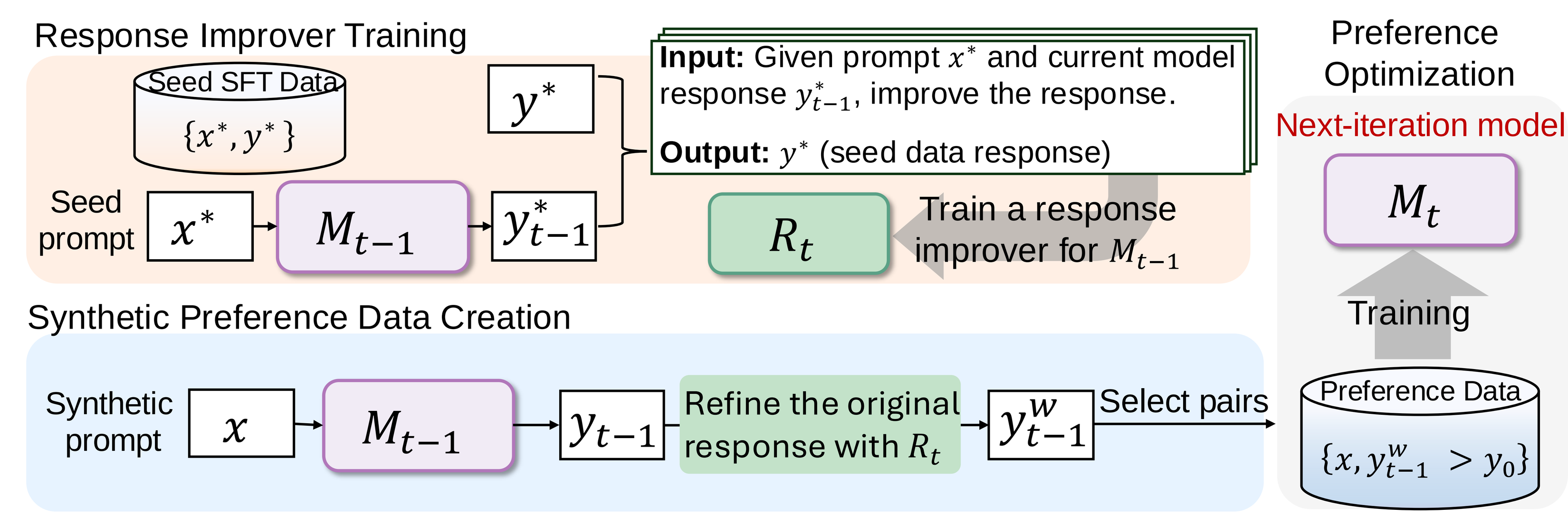

Figure 4. Overview of SynPO in the $\mathbf{t^{th}}$ iteration.

Research Intuition

Intuitions of training LLMs as self-response improvers:

- **Intuition 1:** LLMs excel at identifying distribution gaps between texts[2, 1].
- **Intuition 2:** Refining a response is generally easier than generating a high-quality response from scratch [3].

Response Improver Training

At iteration t, the policy model $\pi_{\theta_{t-1}}$ generates outputs for seed prompts: $\mathbf{y}^*_{(t-1),i}$. Training set: $(\mathbf{x}^*_i, \mathbf{y}^*_{(t-1),i})$ as input, $\mathbf{y}^*_i$ (gold standard) as output. Fine-tune π_{θ_0} to obtain response improver $\mathcal{R}_t$, aligning outputs closer to $\mathbf{y}^*_i$.

Response Improving

For synthetic prompt $\mathbf{x}_i$: (1) Generate $\mathbf{y}_{(t-1),i} \sim \pi_{\theta_{t-1}}(\cdot|\mathbf{x}_i)$. (2) Refine with $\mathcal{R}_t$ to get chosen response $\overline{\mathbf{y}_{(t-1),i}} \sim \mathcal{R}_t(\cdot|\mathbf{x}_i, \mathbf{y}_{(t-1),i})$. (3) Use initial output $\mathbf{y}_{(0),i} \sim \pi_{\theta_0}(\cdot|\mathbf{x}_i)$ as rejected response.

Data Filtering

Filter self-generated data using a small model (e.g., 0.4B PairRM) for scoring. Integrate into synthetic preference data for next iteration.

Synthetic Preference Optimization

Denoting $\mathcal{D}$ as the synthetic preference data, we follow SimPO for training:

$$\theta_t \leftarrow \arg\min_{\theta} \mathbb{E}_{(\mathbf{x}_i, \mathbf{y}^w_i, \mathbf{y}^l_i)\sim\mathcal{D}} \left[\log\sigma \left(\frac{\beta}{|\mathbf{y}^w_i|} \log \pi_{\theta_{t-1}}(\mathbf{y}^w_i \mid \mathbf{x}_i) - \frac{\beta}{|\mathbf{y}^l_i|} \log \pi_{\theta_{t-1}}(\mathbf{y}^l_i \mid \mathbf{x}_i) - \gamma \right) \right]$$

SynPO helps the model learning to improve its own outputs iteratively. The entire optimization process is performed on synthetic data, only a small set for validation.

Results and Discussion

SynPO not only benefits LLM alignment with human preferences, but also improves generalist capabilities across various tasks.

Instruction-following Capability

SynPO significantly improves the instruction-following abilities of LLMs.

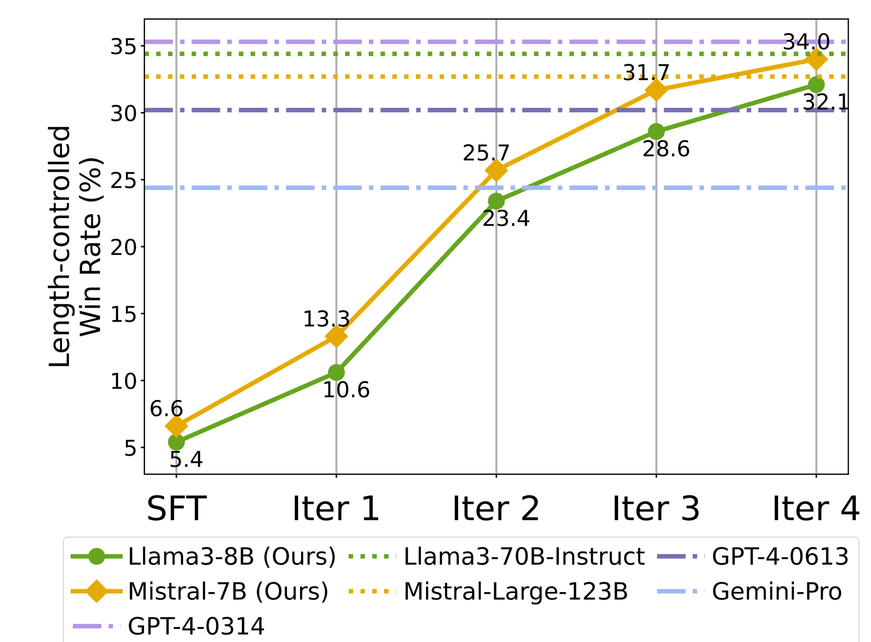

Figure 5. LC Win Rate on AlpacaEval 2.0.

Model		Size	LC (%)	WR (%)
gpt4_1106_preview		-	50.0	50.0
GPT-4 (03/14)		-	35.3	22.1
Meta-Llama-3-70B-Instruct		70B	34.4	33.2
Mistral-Base-SynPO	*Iter4*	7B	34.0	36.4
Mistral Large (24/02)		123B	32.7	21.4
Mistral-Base-SynPO	*Iter3*	7B	31.7	33.8
GPT-4 (06/13)		-	30.2	15.8
Claude 2		-	28.2	17.2
Claude 2.1		-	27.3	17.0
Mistral-Base-SynPO	*Iter2*	7B	25.7	28.1
gemini-pro		-	24.4	18.2
Mixtral-8x7B-Instruct-v0.1		8x7B	23.7	18.3
Mistral-7B-Instruct-v0.2		7B	17.1	14.7
Mistral-Base-SynPO	*Iter1*	7B	13.3	15.3
Mistral-Base-SFT		7B	6.6	3.6

Figure 6. AlpacaEval 2.0 Leaderboard.

General Task Performance

Self-boosted models achieve 3.2% to 5.0% higher average performance than SFT models on Open LLM leaderboard.

Model	Arc	HellaSwag	TQA	MMLU	Winogrande	GSM8k	Average
LLama3-Base-SFT	60.92	81.28	45.37	63.80	76.72	51.93	63.34
Manual Collection	66.72	82.89	59.47	63.10	77.82	45.72	65.95
Sampling-Ranking *Iters**	66.38	82.71	59.84	63.37	77.27	54.40	67.33
Self-Rewarding *Iters**	64.76	82.48	55.54	63.42	77.03	54.59	66.30
SynPO *Iter1*	63.99	82.66	54.20	64.02	77.51	56.10	66.41
SynPO *Iter2*	65.70	83.22	61.73	64.03	76.56	56.25	67.92
SynPO *Iter3*	66.55	83.57	63.53	63.91	76.80	55.27	68.27
SynPO *Iter4*	66.47	83.44	63.69	63.79	76.90	55.72	**68.34**

Table 1. Open LLM Leaderboard results. * represents the best performance across multiple iterations.

Takeaways

- SynPO generates **diverse prompts** to support iterative model improvement.
- Response improver trains via LLM-learned **implicit generative rewards**.
- Small, high-quality validation data **anchors training** and guides better synthetic generation.

[1] Describing differences between text distributions with natural language.
[2] Explaining patterns in data with language models via interpretable autoprompting.
[3] Self-refine: Iterative refinement with self-feedback.