自動プロンプト最適化は個人的選好の予測精度を向上させるか？

劉 弘毅 [1] 谷中 瞳 [1]
[1] 東京大学
{kokiryu, hyanaka}@is.s.u-tokyo.ac.jp

概要

LLM による自動プロンプト最適化は活発に研究されている分野であり，主に人の主観によらず一意に正解が定まるタスクに対する有効性が報告されてきた．しかし，主観的評価によって定められたラベルの予測タスクに対するこれらの手法の効果は検証されていない．本研究では，テキストへの個人的選好の予測タスクに既存の複数の自動プロンプト最適化手法を適用し，精度向上の程度を検証する．検証の結果，(1) 個人的選好の予測タスクについては既存手法による精度の向上が見られないこと，(2) プロンプト書き換えの反復が精度向上に与える効果が小さいこと，(3) LLM が生成した誤りの情報が最適化の過程でうまく参照されていないこと を確認した．

1 はじめに

大規模言語モデル (LLM) を特定のタスクに活用する際には，与えるプロンプトによって精度が大きく変化することが知られている [1]．これを受けて近年活発に研究されているのが**自動プロンプト最適化**の手法である．

自動プロンプト最適化では，特定のタスクに対する最適なプロンプトを LLM を用いて自動で生成させる．既存研究 [2, 3, 4] では主に算数の問題 [5] などの人の主観によらず一意に正解が定まるタスクに対して精度向上の効果が確認されている．

一方で，映画や物語のあらすじに対する個人的選好 [6] などの主観的な評価に基づいて設定されたラベルの予測についても LLM の活用が研究されている．既存の最適化手法ではプロンプトの書き換えが主に LLM の事前知識に基づいて行われていることを指摘する論文 [7] も存在するため，事前知識のみをもとに改善を行うことが難しい主観的ラベルの予測タスクに対してこれらの手法が適用できるかは自明ではない．

個人的選好の予測を行う既存手法 [6] では，LLM を Fine-tuning した評価器に同じ評価者による選好を Few-shot の例示として与えることで精度の向上を達成している．しかし，Fine-tuning による精度向上は新たなドメインに適用するためのコストが高い．また，著者ら自身によって Few-shot の例示の数の増加による精度の向上の限界が示唆されており，精度向上のための新たな手法が必要とされている．

そこで本研究では，既存の複数の自動プロンプト最適化手法を個人的選好の予測に適用し，その効果を検証する．本研究による主な貢献は以下の通りである．

- 既存手法では個人的選好の予測に対する精度の向上が難しいことを明らかにした．
- 個人的選好の予測においてはプロンプトの書き換えの反復による精度向上の効果が小さいことを示した．
- LLM による誤り訂正の過程を検証し，誤りから得られる情報が適切にプロンプトに反映されていないことを確認した．

2 自動プロンプト最適化

2.1 今回用いる手法

Zhou ら [2] は，あるタスクに属する問題 Q，正解ラベル A，タスクを解くためのプロンプト p に対し，(p, Q) を入力として LLM から得られた出力の A に対する評価値 $f(p, Q, A)$ を用いて，以下のような最適化問題として自動プロンプト最適化を定式化している．

$$p^* = \arg\max_p \mathbb{E}_{(Q,A)}\left[f(p, Q, A)\right] \quad (1)$$

今回の実験では，これを達成する既存手法として Iterative-APE [2] 及び APO [3] を主に取り扱う．これらの手法による主なステップを以下に示す．

2.1.1 プロンプトの初期化

まず，最適化の出発点となるプロンプトの集合 P_0 を得る．今回は APO での設定に則り，タスクごとに固定のプロンプトをこの初期プロンプトとして用いる．

2.1.2 新規プロンプトの提案

次に，現在のプロンプトの集合 P_i からプロンプト $p \in P_i$ を取り出し，これを適当な変換 T を用いて書き換えることで新たなプロンプトの候補 $p' = T(p)$ を得る．Iterative-APE，APO によるプロンプトの書き換えは以下のように行われる．

意味を保った書き換え (Iterative-APE) “Generate a variation of the following instruction while keeping the semantic meaning.” というプロンプトを用いて，既存のプロンプトの意味を保ちながら言い換えを行う．

誤りの訂正 (APO) 単なるプロンプトの言い換えではなく，現在のプロンプトによって生じている誤りを明示的に示し，それを基にプロンプトの書き換えを行う．

まずプロンプト $p \in P_i$ に対して，p による出力が誤りを含むような入出力の例 $E = \{(Q_1, A_1), ..., (Q_k, A_k)\}$ をいくつかサンプリングする．その上で，LLM に p による誤りの原因を指摘させる指示 Δ を与え，誤りの原因 g をテキストとして出力させる．

$$g = LLM_{\Delta}(p, E) \tag{2}$$

次に，E, g をもとに p を修正させる指示 δ を用いて，新しいプロンプト p' を提案させる．

$$p' = LLM_{\delta}(p, g, E) \tag{3}$$

2.1.3 最適プロンプトの探索

これらの書き換えによって，プロンプトの集合が $P'_i = P_i \cup \{T(p) \mid p \in P_i\}$ に拡張される．ここから精度を向上させる可能性のあるプロンプトを選別することで，次の書き換えに用いるプロンプトの集合 $P_{i+1} \subset P'_i$ を得る．

今回はバリデーション用に抽出した小規模な訓練用データセット $\widetilde{D}_v$ に対する $p \in P'_i$ の精度を計測することで，精度の高いプロンプトを選択する．

以上を繰り返すことで，最適化されたプロンプト p^* が得られる．

2.2 その他の既存手法

その他にも，誤り訂正に基づく自動プロンプト最適化の手法として ABO [7] 及び PE2 [4] などが存在する．これらの手法は Iterative-APE 及び APO よりも後発の手法であるが，初期プロンプトや入力長に対する制限が存在するため，今回は対象としない．また，どちらも Iterative-APE/ABO がある程度有効であったタスクに対する改善を行なった論文であり，今回の分析には Iterative-APE/ABO の適用で十分である．

3 個人的選好の予測精度の検証

ここからは，既存の自動プロンプト最適化手法を個人的選好の予測タスクに適用し，精度に対する影響を検証する．

3.1 実験設定

3.1.1 検証に用いるデータセット

今回は，Wang ら [6] によって作成された 2 種類のデータセットである Per-MPST および Per-DOC を用いて検証をおこなう．また比較対象として，先行研究の評価対象である GSM8K [5] (小学生レベルの算数問題のデータセット) を用いた評価も行う．

Per-MPST と Per-DOC はどちらもテキストに対する個人的選好を予測するデータセットである．Per-MPST では入力が映画のあらすじ，出力が自由記述によるレビュー及び 10 段階の点数となる．Per-DOC では入力が 2 つの文章，出力が評価者がより好む文章のラベルとなる．評価の際には，入力に加えてその評価者による過去の評価 k 例 (Per-MPST では $k = 3$，Per-DOC では $k = 1$) が与えられる．

全てのデータセットについて，元の訓練用データセット D_t から

- プロンプト書き換え用訓練データセット $\widetilde{D}_t$
- 最適プロンプト探索用検証データセット $\widetilde{D}_v$

を構築し，プロンプト最適化に用いた．評価は元のデータセットの評価データ全件を用いて行なった．Per-MPST 及び Per-DOC の評価データに関する統計情報を付録 A.1.1 に記載する．

Per-MPST については，生成されたプロンプトの頑健性を調べるため，元の評価データ (Original) に加え，評価者ごとの点数の分散が大きく予測が難しいと思われる 100 件のデータからなる評価データセット (Hard) も構築し，評価に用いた．

3.1.2 検証対象とする自動プロンプト最適化手法

今回はタスクごとに固定のプロンプトテンプレートを定め，その冒頭にある Instruction 部分に既存の自動プロンプト最適化手法を適用し，それによる精度の変化を分析した．利用したプロンプトテンプレートは付録 A.1.2 に記載する．

初期プロンプトとしてはタスクごとに先行研究に準じた固定のものを利用した．詳細は付録 A.1.3 に記載する．

新規プロンプトの提案については Iterative-APE [2]，及び APO [3] のものを用い，結果を比較した．また，APO の誤り訂正時に Few-shot の例示も入力に与える手法を APO-Demo として比較に加えた．書き換えに用いるプロンプトとしては，各論文に記載のものを基に PE2 [4] 内での比較実験を参考に入出力形式に関する指定を加えて使用した．プロンプトを生成する際の LLM の出力トークン長としては 400 トークンを指定した．

いずれの手法についても，各ステップで候補として残るプロンプト $p \in P$ ごとに 4 回ずつ書き換えを行い，これらを最適プロンプトの探索に利用した．新規プロンプトの提案及び最適プロンプトの探索からなるステップを 3 回繰り返し，得られたプロンプトで $\widetilde{D}_v$ に対する精度が最良のものを評価データに適用し，最適化前の基準値 (Vanilla) と比較した．

$\widetilde{D}_v$ 及び最終的な評価データに対して精度を計測するための指標としては，GSM8K 及び Per-DOC については正答率，Per-MPST については Spearman の相関係数を用いた．

3.1.3 検証に用いた LLM

検証にあたっては，Llama 3.1 8B Instruct [8] 及び GPT-4o [9] を LLM として用いた．

3.2 実験結果

Llama 3.1 8B Instruct を用いた主な実験結果を表 1，2 に示す．

GSM8K に対する実験では，すべての手法で大幅に正答率が向上し，誤り訂正に基づく書き換えである APO で特に大きい正答率の向上が見られた．この結果は先行研究で示されたものと合致する．

表 1 各手法の評価用データに対する正答率

	最適化手法			
データセット	Vanilla	Iterative-APE	APO	APO-Demo
GSM8K	0.42	0.57	**0.75**	N/A
Per-DOC	0.48	0.48	**0.51**	0.44

表 2 各手法の Per-MPST に対する相関係数

	最適化手法			
評価データ	Vanilla	Iterative-APE	APO	APO-Demo
Original	0.25	0.24	0.22	**0.26**
Hard	**0.28**	0.23	0.07	0.03

一方で Per-DOC については, いずれの手法でもランダムより有意に良い正答率は得られなかった．また Per-MPST については，Original 評価データに対する性能ではほとんど変化が見られず，Hard 評価データに対しては APO / APO-Demo で大幅に性能が低下するという結果が得られた．

このことから，既存の自動プロンプト最適化手法は個人的選好の予測精度の向上に有効に寄与しない可能性が示唆される．

また，GPT-4o についても Llama 3.1 8B Instruct に対する結果と同様の傾向が見られた．結果は付録 A.2.1 に示す．

3.3 結果に対する考察

3.3.1 プロンプト最適化過程の検証

GSM8K と Per-MPST における APO 及び APO-Demo の最適化の各ステップにおいて，

- そのステップで得られたすべてのプロンプトの検証データに対する精度
- そのステップまでで得られた最良のプロンプトの評価データに対する精度

をプロットしたものを図 1 に示す．ここから，最適化の過程について以下のことがわかる:

- プロンプト書き換えの反復による精度向上の効果は小さい．精度の大幅な向上が見られる GSM8K について，精度の大幅な変化は最初のプロンプト (“Let’s think step by step”) の一回の書き換えのみで行われている．各ステップの書き換えにおいて精度の低いプロンプトも多く生成

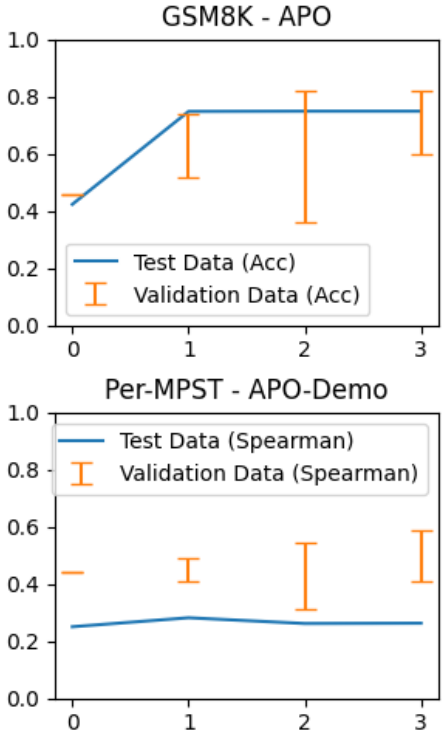

図 1　各ステップで得られたプロンプトの精度の変化

されていることも読み取れる．

- 個人的選好の予測タスクについては，少量の検証データに対する精度をもとに最良プロンプトを選択することは適切でない．Per-MPST に対する実験では，そもそも検証データに対する精度の上昇と同等の精度の上昇が評価データに対しては見られなかったことがわかる．少量の検証データをとり，それに対する精度をもとに最良のプロンプトを選択する」という手法が明示的な正解が存在しないタスクについてはうまくはたらかない可能性が示唆される．

3.3.2　誤り訂正の効果の検証

APO 及び APO-Demo において，誤りに基づいてプロンプトを一回書き換えた際の GSM8K の検証データに対する精度の変化を図 2 に示す．また，Per-MPST に対する実際の誤り訂正の際の誤りの原因を指摘する中間出力を付録の図 4 に示す．ここから，プロンプトの誤り訂正の効果について以下のことが読み取れる:

- 誤り訂正の前後で精度が上昇しない場合が多く観測される． 図 2 では，GSM8K のような比較的大きい精度の向上が見られるタスクに対しても，誤り訂正の前後でプロンプトの精度が低下

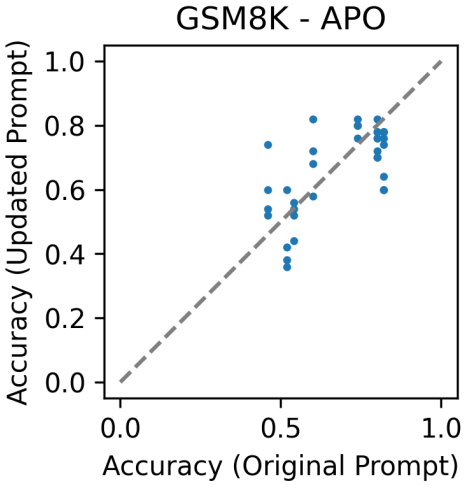

図 2　プロンプトの誤りに基づく書き換えの前後における精度の変化

してしまう例が向上する例と同程度みられる．

- 誤り訂正の内容は誤りの観察ではなく LLM の事前知識に基づいている． 図 4 で指摘されている誤りの原因は，プロンプトの不備のみを指摘し実際の誤りの内容に基づかない項目，そして「訓練データの増加」などのプロンプト最適化で達成できない項目から構成されていることがわかる．このことから，誤り訂正が LLM の事前知識のみに基づいて行われていることが示唆される．Ma ら [7] も BIG-bench Hard [10] に対する実験で同様の指摘をしており，これが再確認されたといえる．

3.3.1，3.3.2 で示されたこれらの原因によって，個人的選好の予測タスクにおいて自動プロンプト最適化による恩恵が得られなかったと考えられる．

4　おわりに

本研究では，テキストに対する個人的選好の予測タスクについて，既存の自動プロンプト最適化手法による精度向上が見られないことを確認した．また，最適化の途中経過の観察を通して

- プロンプトの書き換えの反復による精度向上の効果が小さいこと
- LLM が生成した誤りから得られる情報が適切に書き換えられたプロンプトに反映されていないこと

が，効果が見られなかった原因であると考察した．誤り訂正についてはプロンプトの工夫等で改善できる可能性があり，これは残された課題である．

謝辞

本研究は JST さきがけ JPMJPR21C8 の支援を受けたものである.

参考文献

[1] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. **Advances in neural information processing systems**, Vol. 35, pp. 22199–22213, 2022.

[2] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In **The Eleventh International Conference on Learning Representations**.

[3] Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**, pp. 7957–7968, 2023.

[4] Qinyuan Ye, Mohamed Ahmed, Reid Pryzant, and Fereshte Khani. Prompt engineering a prompt engineer. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, **Findings of the Association for Computational Linguistics: ACL 2024**, pp. 355–385, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[5] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. **arXiv preprint arXiv:2110.14168**, 2021.

[6] Danqing Wang, Kevin Yang, Hanlin Zhu, Xiaomeng Yang, Andrew Cohen, Lei Li, and Yuandong Tian. Learning personalized alignment for evaluating open-ended text generation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, **Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing**, pp. 13274–13292, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[7] Ruotian Ma, Xiaolei Wang, Xin Zhou, Jian Li, Nan Du, Tao Gui, Qi Zhang, and Xuanjing Huang. Are large language models good prompt optimizers? **arXiv preprint arXiv:2402.02101**, 2024.

[8] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. **arXiv preprint arXiv:2407.21783**, 2024.

[9] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. **arXiv preprint arXiv:2410.21276**, 2024.

[10] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, **Findings of the Association for Computational Linguistics: ACL 2023**, pp. 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics.

[11] jinja2. Online; Original date: 2008-06-09, 5 2024. A very fast and expressive template engine.

A 付録

A.1 実験設定の詳細

A.1.1 Per-MPST/Per-DOC の統計情報

各評価データに関する統計情報を表 3 に記載する．入力長/レビュー長は平均文字数を表す．

表 3 評価データの統計情報

データセット	件数	入力長	レビュー長
Per-MPST	920	1265.4	752.1
Per-DOC	954	7263.2	N/A

A.1.2 実験に用いたプロンプトの形式

実験に際しては，APO‘ [3] の設定を参考に，定められたテンプレートの冒頭に記載された Instruction の最適化を行なった．利用したフォーマットを Jinja2 [11] の方式で表 3 に示す． このテンプレートの中で，“instruction” のみを最適化の対象とした．“output_format” および “final_instruction” についてはデータセットごとに固有の人手で書いたテキストを用いた．“exmples” 及び “input” については各評価データを整形したものを用いた．

```
# Task
{{ instruction }}
# Output format
{{ output_format }}
{% if examples is not none %}
# Examples
{{ examples }}
{% endif %}
# Prediction
{% if final_instruction is not none %}
{{ final_instruction }}
{% endif %}Input: {{ input }}
```

図 3 最適化用テンプレート

A.1.3 初期プロンプト

GSM8K に対する初期プロンプトとしては，PE2 [4] での設定と同様 “Let’s think step by step” を用いた．Per-MPST については Wang ら [6] の提案手法で利用されたプロンプトをそのまま用いた．Per-DOC については，提案手法で用いられていたプロンプトが「冒頭にタスクの説明を記載する」という今回のフォーマットと合致しなかったため，Per-MPST に対して用いたプロンプトの表現を一部書き換えることによって初期プロンプトを得た．

A.2 その他の実験結果

A.2.1 GPT-4o での結果

GPT-4o に対して得られた結果を表 4，5 に示す．Llama 3.1 8B Instruct による実験と同様，最適化 (APO-Demo) による精度向上の効果が表れていないことがわかる．

表 4 各手法の評価用データに対する正答率

	最適化手法	
データセット	**Vanilla**	**APO-Demo**
GSM8K	0.95	**0.96**
Per-DOC	**0.48**	0.46

表 5 各手法の Per-MPST に対する相関係数

	最適化手法	
評価データ	**Vanilla**	**APO-Demo**
Original	0.37	**0.40**
Hard	**0.28**	0.26

A.2.2 LLM の誤り訂正の例

Here are three reasons why the prompt could have gotten these examples wrong:
1. **Lack of clear criteria for tone and style analysis**: The prompt asks the model to analyze the critic’s tone and style, but it doesn’t provide clear criteria for what constitutes a ”good” or ”bad” tone and style. This makes it difficult for the model to accurately identify the critic’s tone and style.
2. **Insufficient context for the critic’s preferences**: The prompt doesn’t provide enough context about the critic’s preferences, such as their favorite genres, directors, or actors. This makes it challenging for the model to accurately infer the critic’s preferences and write a review that reflects their style.
3. **Inadequate training data**: The prompt’s training data may not be representative of the critic’s writing style or preferences. The examples provided may not be sufficient to capture the critic’s tone, style, and preferences, leading to inaccurate analysis and review generation.

図 4 誤り訂正のために Llama 3.1 8B が行なった出力