検索付き拡張生成における ハイパーパラメータとプロンプトの同時最適化

鈴木海渡 [1]　水野尚人 [1]　柳瀬利彦 [1]　佐藤元紀 [1]
[1] 株式会社 Preferred Networks
{kaitos,naotomizuno,yanase,sato}@preferred.jp

概要

Retrieval-augmented generation（RAG）は，コーパスからの情報検索を活用することにより，大規模言語モデル（LLM）の性能を大幅に向上させる．RAG の性能は，RAG パイプラインにおけるハイパーパラメータと，LLM に与えるプロンプトに大きく依存する．本研究で提案する UniOptRAG フレームワークでは，単変量最適化アルゴリズムを組み合わせることで，ハイパーパラメータとプロンプトを同時に最適化する．この同時最適化によっていずれか一方のみを最適化する場合の性能を上回り，既存研究でファインチューニングや高度な推論パイプラインを用いた場合と同等の性能を達成することを示す．

1　はじめに

近年 LLM はさまざまなタスクにおいて顕著な性能を示している．さらに，外部の文書を組み込む RAG の登場によって，LLM はより広範な知識をもとに文章を生成することができるようになった [1]．

典型的な RAG は，Retriever と Generator の 2 つの要素から構成される．まず，Retriever はクエリに関連する文書を外部のコーパスから検索する．次に，Generator は，クエリと検索結果上位の文書，タスクの説明を組み合わせて LLM に与えるプロンプトを構築し，それを LLM に与えて出力を生成する．

RAG は高い性能を示す一方で，その設計には依然として困難が伴う．例えば，以下のような非自明な問いがしばしば生じる．

- Retriever のパラメータをどう決めるか？
- LLM に検索結果をいくつ入力するか？
- LLM の応答を制御するためのプロンプトはどのように設計するのが望ましいのか？

これらのハイパーパラメータやプロンプトの決定は，RAG パイプライン全体の性能に大きく影響するため，慎重な検討が必要である [2, 3]．

本研究は，RAG パイプラインにおけるハイパーパラメータとプロンプトを同時に最適化する試みである．提案する UniOptRAG フレームワークでは，単変量最適化アルゴリズムを組み合わせて RAG パイプラインを最適化する．

本研究の主な貢献は以下のとおりである．

1. RAG のハイパーパラメータとプロンプトを同時に最適化するフレームワークを提案する．
2. 実験を通じて，提案手法による同時最適化が RAG パイプラインの性能を大幅に向上し，他の RAG 手法と比較して競争力のある性能を達成することを示す．
3. 最適なハイパーパラメータとプロンプトがタスクに強く依存することを明らかにし，本フレームワークによる同時最適化の重要性を示す．

2　提案手法

本研究では，RAG のハイパーパラメータとプロンプトを同時に最適化するフレームワークとして，UniOptRAG を提案する．

ハイパーパラメータとプロンプトの同時最適化においては，数値やカテゴリカルな探索空間とテキストの探索空間の両方を同時に扱う必要がある．このように複雑な探索空間では，変数同士の相互作用を考慮することも困難である．

UniOptRAG フレームワークは，単変量ハイパーパラメータ最適化アルゴリズム [4] を拡張する．このアルゴリズムは変数間の相互作用を無視し，それぞれを独立に扱う．これにより，ハイパーパラメータ最適化（HPO）アルゴリズムに対し，プロンプト最適化のための遺伝的アルゴリズムなど，他のアルゴリズムを組み合わせることが可能となる．

i 次元目のハイパーパラメータを θ_i とし，精度などの評価スコアと組になった過去の i 次元目のハイ

Algorithm 1 UniOptRAG フレームワークの概要

Require: $\mathcal{S}_\theta$ (HPO algorithm), $\mathcal{S}_p$ (prompt optimization algorithm)
Ensure: $\mathcal{H}_i$ (i-th dimension hyperparameter history), $\mathcal{H}_p$ (prompt history)
  for each hyperparameter index i **do**
    $\mathcal{H}_i \leftarrow \phi$
  end for
  $\mathcal{H}_p \leftarrow \phi$
  while termination condition is not satisfied **do**
    for each hyperparameter index i **do**
      $\theta_i \leftarrow \mathcal{S}_\theta(\mathcal{H}_i)$
    end for
    $p \leftarrow \mathcal{S}_p(\mathcal{H}_p)$
    $s \leftarrow \text{evaluate}(\theta_1, \ldots, \theta_N, p)$
    for each hyperparameter index i **do**
      $\mathcal{H}_i \leftarrow \mathcal{H}_i \cup \{(\theta_i, s)\}$
    end for
    $\mathcal{H}_p \leftarrow \mathcal{H}_p \cup \{(p, s)\}$
  end while

パーパラメータ値の履歴を $\mathcal{H}_i$ とする．新たな θ_i を提案する際，単変量 HPO アルゴリズムは $\mathcal{H}_i$ に含まれるハイパーパラメータと評価スコアのペアだけを用い，$\mathcal{H}_j \mid j \neq i$ は必要としない．

プロンプトを表す変数を p，履歴を $\mathcal{H}_p$ とする．プロンプトも単変量であると仮定し，最適化アルゴリズムは $\mathcal{H}_p$ のみの情報に基づいて p を提案する．

提案する UniOptRAG フレームワークの概要を Algorithm 1 に示す．ハイパーパラメータを提案するアルゴリズム $\mathcal{S}_\theta$ と，プロンプトを提案するアルゴリズム $\mathcal{S}_p$ は共に単変量最適化アルゴリズムなので，それぞれの履歴は互いに独立である．

3 実験

3.1 RAG アーキテクチャ

実験では，以下のモジュールから構成される RAG アーキテクチャを検討する．なお，ここで示す RAG の構成は一例であり，提案する UniOptRAG フレームワークは任意の RAG 構成に適用可能である．

(1) Retriever: 各データに含まれる文をクエリとして扱い，事前に構築されたコーパスに対して検索を行う．まず BM25 検索 [5] を用いて検索を実行し，その結果をリランキングする．リランキングにおいては，BM25 検索のスコアと，クエリと検索結果の埋め込みベクトル間のコサイン類似度であるベクトルスコアを組み合わせて算出するハイブリッドスコアをベースに，検索結果の並び替えを行う．

(2) Generator: 上位の検索結果を，タスクの説明を含むプロンプトテンプレートに挿入することでプロンプトを構築する． このプロンプトを LLM に入力し，推論を行って出力を生成する．

実験では，次の変数を最適化対象とする．

1. `hybrid_method`: 検索結果を再ランキングする際に，BM25 スコアとベクトルスコアをどのように組み合わせるかを指定する．“wsum” の場合，BM25 スコアとベクトルスコアをそれぞれ最大値が 1 になるように正規化し，それらの重み付き平均をハイブリッドスコアとする．“rrf” の場合，Reciprocal Rank Fusion（RRF) [6] に基づいてハイブリッドスコアを計算する．
2. `fusion_weight`: BM25 スコアとベクトルスコアを組み合わせる際に使用される重み．BM25 スコアは `fusion_weight` で，ベクトルスコアは `(1-fusion_weight)` でそれぞれ乗算され，それらの和がハイブリッドスコアとなる．
3. `rrf_k`: RRF の挙動を制御するパラメータ．
4. `num_results`: 検索結果の件数．
5. `reorder_method`: 検索結果をプロンプトに組み込む際に，どのように並べ替えるかを指定する．“normal” はスコアが高い順に並べ，“inverse” はスコアが低い順に並べる．“lost_in_the_middle” [7] は，最も関連度の高い情報を入力の冒頭または末尾に配置する．
6. `prompt_template`: LLM に入力される，対象タスクの説明を含んだ文章．
7. `temperature`: LLM の温度パラメータ．

3.2 データセット

本研究では，3 つのデータセットで実験を行った．最適化の目的関数および評価指標として，選択問題の正答率を用いた．最適化には，各データセットの訓練データから 500 例をサンプリングして用いた．

StrategyQA [8] は，オープンドメイン質問応答のベンチマークである．IAG [9] に従い，訓練データの 4 分の 1 を検証用データとして評価に用いた．**AI2 Reasoning Challenge (ARC)** [10] は，科学系の試験問題を含む多肢選択形式の質問応答ベンチマークである．評価には公式のテストデータを用いた．**TriviaQA** [11] は，読解問題およびオープンドメイン質問応答のベンチマークである．Self-RAG [12] に従い，検証データを評価に用いた．

3.3 実験設定

Retriever の一部として，OpenAI の text-embedding-ada-002 モデルを使用し，ベクトルスコアを算出するためのテキスト埋め込みを計算した．

RAG パイプラインで使用する LLM には，llama2-70B-chat [13] モデルを採用し，GPTQ 量子化 [14] を用いて 4 ビットに量子化した．

ハイパーパラメータの最適化には，Optuna [15] に実装されている Tree-structured Parzen Estimator (TPE) [4] アルゴリズムを使用した．

プロンプト最適化手法として，先行研究で提案された遺伝的アルゴリズムベースの方法を採用した [16]．この手法では，最近のプロンプト最適化手法 [17, 18, 19] をベースとし，トーナメント選択を用いて親個体となるプロンプトを 2 つ選択し，交叉と突然変異によって次世代個体となる新たなプロンプトを生成する．交叉と突然変異は，LLM を用いて既存のプロンプトを言い換えることで実装した．このための LLM としては GPT-3.5 モデルを採用した．

最適化にあたっては，500 回の試行を行った．各実験は 3 回実施し，平均値と標準誤差を求めた．

3.4 実験結果

表 1 3 つのタスクにおける実験結果を示す．太字の値は公開モデルの中で最良の性能を示し，灰色の太字で示されている値は，非公開モデルも含めた最良の性能を示す．

	Method	StrategyQA	ARC	TriviaQA
非公開モデル	ChatGPT	52.0	75.3	**74.3**
	Ret-ChatGPT	-	75.3	65.7
	IAG-GPT	**76.2**	-	-
公開モデル	IAG-Student	66.6	-	-
	Self-RAG$_{7B}$	-	67.3	66.4
	Self-RAG$_{13B}$	-	73.1	**69.3**
本研究	Default	60.1	57.7	58.8
	Hyperparameter-only	62.6 ± 0.2	72.7 ± 0.1	65.2 ± 0.2
	Prompt-only	65.7 ± 0.9	74.0 ± 0.5	63.3 ± 0.4
	Joint	**68.5** ± 0.8	**75.4** ± 0.7	66.1 ± 0.5

実験結果を表 1 に示す．StrategyQA データセットに関しては，先行研究の結果として IAG の結果を参照する．IAG は帰納的知識を提供する追加コンポーネントであるインダクターを組み込み，QA ベンチマークにおいて顕著な性能を達成している．

ARC および TriviaQA データセットに対しては，Self-RAG の結果と比較する．Self-RAG はファインチューニングを通じて導入した特殊なトークンを用いて RAG プロセスを強化するフレームワークであり，QA ベンチマークにおいて優れた性能を示している．また，参考として ChatGPT と Ret-ChatGPT (検索機能を組み込んだ ChatGPT) の結果も提示する．

本研究における結果の比較として，異なる最適化戦略を用いた結果を提示する． Default は，デフォルトのハイパーパラメータ値およびプロンプトテンプレートを用いた結果を指す．Hyperparameter-only および Prompt-only は，それぞれハイパーパラメータとプロンプトテンプレートのみを最適化した際の結果を示す．Joint は，本研究で提案する同時最適化によって得られた結果を示す．

全てのデータセットにおいて, Hyperparameter-only と Prompt-only は Default と比較して顕著に性能を改善した．さらに，同時最適化はさらに性能を向上させ，本研究における結果の中で一貫して優れた性能を達成した．これは，より良い性能を得るためにはハイパーパラメータとプロンプトテンプレートの両方の最適化が重要であり，提案する同時最適化が両者を効果的に最適化することを示している．

同時最適化の結果は，非公開モデルの LLM を用いた先行研究と比較しても競争力のある性能を達成している．これは，比較的単純な RAG アーキテクチャにおけるハイパーパラメータとプロンプトテンプレートの最適化が，追加コンポーネントやファインチューニングを活用した手法にも匹敵する性能を達成するほど強力であることを示している．

3.5 結果の分析

図 1 は，高性能な試行における各ハイパーパラメータの分布を示している．ある特定の値の確率が高いということは，高い性能を得るためにその値を採用することが重要であることを示している．

特に `hybrid_method`, `num_results`, `reorder_method` のようなハイパーパラメータに顕著であるように，各ハイパーパラメータに対して特定の値が強く好まれる傾向が見られる．これらの好まれる値はデータセットごとにしばしば異なるため，各データセットに対してハイパーパラメータをチューニングすることの重要性を示唆している．

`fusion_weight` パラメータは小さい値を取ることが多く，これはベクトルスコアを重視していることを意味する．しかし，BM25 スコアにも一定の重みが割り当てられており，両方の検索手法が活用されていることを示している．`num_results` パラメー

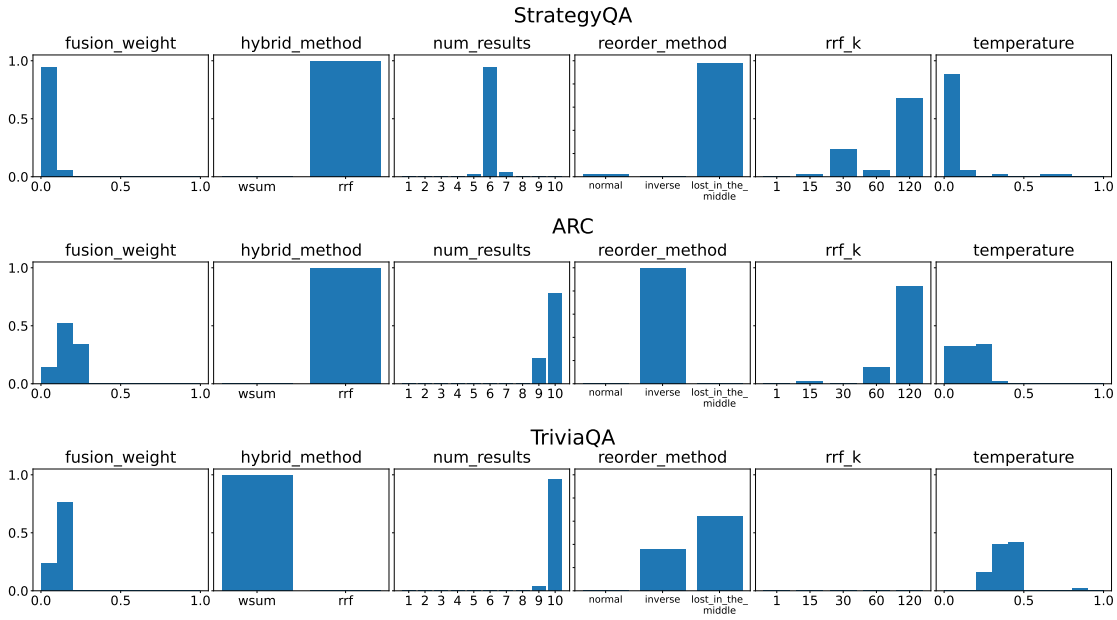

図 1　高性能な試行におけるハイパーパラメータの分布．各図は最適化中の上位 10%の試行についてのヒストグラム．

タは大きな値を取ることが多く，幅広い検索結果を考慮することの重要性を示唆している．ただし，StrategyQA では 6 に収束し，「多ければ良い」というわけでもないことがわかる．`reorder_method` パラメータに関しては，“inverse” や “lost_in_the_middle” の値が好まれ，検索結果を適宜並び替えることに一定の重要性があることが示されている．

付録 B にデフォルトのプロンプト，付録 C に最適化されたプロンプトを示す．最適化されたプロンプトでは，より詳細で説明的なタスクの指示が含まれていることが確認できる．これらの記述は人間から見た意味合いに大きな差はないが，RAG パイプラインの性能に大きな影響を与えており，プロンプトテンプレートの最適化が RAG パイプラインの性能向上において重要であることを示している．

4　関連研究

RAG パイプラインの性能を向上させるために，さまざまな工夫が提案されてきた．IAG [9] は，帰納的知識を提供するインダクタモジュールを導入する．Self-RAG [12] は，ファインチューニングを通じて，文書の取得などを動的に実行するための特別なトークンを導入する．

RAG におけるハイパーパラメータの影響に関する研究も行われている．Same Task, More Tokens [2] は，長い文脈入力を処理することによるデメリットを指摘しており，それが RAG の性能を低下させる場合があると論じている．別の観点として，Lost in the Middle [7] は，取得結果を提示する順序の影響を調査し，最も関連する情報が先頭または末尾に配置されると最良の性能が得られることを示唆した．

プロンプト最適化は，LLM の性能を高めるために近年盛んに研究されている分野の 1 つである．APE [17] は，同義文を生成する LLM を用いて言い換えを行い，最適化の履歴から良いスコアを得られたプロンプトを選択する手法である．OPRO [18] は，最適化履歴で高スコアを得た複数のプロンプトとそのスコアを列挙し，それらを LLM で言い換えて新たなプロンプトを生成する．Promptbreeder [19] では，遺伝的アルゴリズムに基づく手法を用い，LLM を用いて実現される突然変異や交叉などのオペレータを適用して新たなプロンプトを生成する．

5　おわりに

本研究では，ハイパーパラメータとプロンプトを同時に最適化する RAG 最適化フレームワークである， UniOptRAG を提案した．

本研究の実験結果から，ハイパーパラメータとプロンプトを最適化するだけで RAG パイプラインの性能を大幅に向上できることが示された．さらに，異なるタスクでは最適なハイパーパラメータやプロンプトが異なることが明らかとなり，タスクごとの最適化の重要性が示唆された．

今後の研究の方向性として，本フレームワークのさらなる可能性を検証するために，他の RAG 改善手法との組み合わせを検討するなどが考えられる．

謝辞

本研究において議論に参加して頂いた渡邊修平さん，鈴木渓太さん，林浩平さんに感謝いたします．

参考文献

[1] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In **Advances in Neural Information Processing Systems**, 2020.

[2] Mosh Levy, Alon Jacoby, and Yoav Goldberg. Same task, more tokens: the impact of input length on the reasoning performance of large language models. **CoRR**, Vol. abs/2402.14848, , 2024.

[3] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In **Advances in Neural Information Processing Systems**, Vol. 35, pp. 22199–22213, 2022.

[4] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In **Advances in Neural Information Processing Systems**, Vol. 24, 2011.

[5] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In **Proceedings of the Third Text REtrieval Conference (TREC)**, Vol. 500-225, pp. 109–126, 1994.

[6] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In **SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval**, pp. 758–759, 2009.

[7] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. **Transactions of the Association for Computational Linguistics (TACL)**, 2023.

[8] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies. **Transactions of the Association for Computational Linguistics (TACL)**, 2021.

[9] Zhebin Zhang, Xinyu Zhang, Yuanhang Ren, Saijiang Shi, Meng Han, Yongkang Wu, Ruofei Lai, and Zhao Cao. IAG: Induction-augmented generation framework for answering reasoning questions. In **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**, pp. 1–14, 2023.

[10] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. **CoRR**, Vol. abs/1803.05457, , 2018.

[11] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In **Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics**, 2017.

[12] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In **The Twelfth International Conference on Learning Representations**, 2024.

[13] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. **CoRR**, Vol. abs/2307.09288, , 2023.

[14] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training compression for generative pretrained transformers. In **The Eleventh International Conference on Learning Representations**, 2023.

[15] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**, p. 2623–2631, 2019.

[16] 水野尚人, 柳瀬利彦, 佐野正太郎. 自動プロンプト最適化のソフトウェア設計. 言語処理学会第 30 回年次大会 (NLP2024), 2024.

[17] 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**, 2023.

[18] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In **The Twelfth International Conference on Learning Representations**, 2024.

[19] Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. **CoRR**, Vol. abs/2309.16797, , 2023.

A　最適化対象のパラメータの詳細

表 2　最適化対象のパラメータのデフォルト値と探索範囲

パラメータ	デフォルト値	探索範囲
hybrid_method	“wsum”	{“wsum”, “rrf”}
fusion_weight	1.0	[0.0, 1.0]
rrf_k	60	{1, 15, 30, 60, 120}
num_results	5	{1, 2, 3, ..., 10}
reorder_method	“normal”	{“normal,” “inverse,” “lost_in_the_middle”}
prompt_template	付録 B 参照	自由形式の文章
temperature	0.0	[0.0, 1.0]

B　デフォルトのプロンプトテンプレート

表 3　デフォルトのプロンプトテンプレート（StrategyQA の例．他のデータセットでもほぼ同様のものを用いた）

Utilizing the search results, answer the following question with True or False. Please answer with the True or False only, without adding any extra phrase or period.
Search Results
{results}
Examples
{examples}
Problem
Question: {question}
Answer:

C　最適化後のプロンプトテンプレート

表 4　最適化後のプロンプトテンプレート

StrategyQA

To validate the correctness of the given claim, please refer to the search results that have been provided. Your answer should exclusively comprise of the terms "True" or "False" and should abstain from incorporating any extra words or punctuation marks.
Search Results {results}
Examples {examples}
Problem
Question: {question}
Answer:

ARC

To proceed, kindly select the correct response from the search results that are currently being shown. You have the opportunity to choose from four options (A, B, C, and D). Please indicate your choice by using only the uppercase letter and refrain from adding any extra words or punctuation marks.
Search Results {results}
Examples {examples}
Query
Question: {question}
Answer:

TriviaQA

Please employ the collected search data to produce a response to the designated query. It is of utmost importance that your answer is succinct and devoid of any unnecessary words or punctuation marks.
Search Results {results}
Examples {examples}
Aim
Question: {question}
Answer: