コストと精度を考慮した質問と文書に基づく適応的 RAG 選択

穴口 史将 [1,2]　チャクラボルティ シュデシナ [1]　森田 武史 [1,2]
太田 葵 [2]　浅田 真生 [2]　江上 周作 [2]　鵜飼 孝典 [2,3]　濱崎 雅弘 [2]
[1] 青山学院大学　[2] 産業技術総合研究所　[3] 富士通株式会社
morita@it.aoyama.ac.jp　masahiro.hamasaki@aist.go.jp

概要

大規模言語モデルの応答品質の向上を目的として，多様な Retrieval-Augmented Generation（RAG）が研究されている．例えば，知識グラフや要約を用いて外部知識を活用する RAG がある．一方で，単純な質問には NaiveRAG で実用的な精度を維持できる．高度な RAG はコストを増大させ，実運用における効率を低下させるため，入力に応じた RAG 構成の選択が必要である．提案手法は，質問と文書から抽出した特徴量を用いてランキングモデルが性能指標の順位を予測し，ユーザが設定した重みに基づいて適応的な構成を選択する．評価実験の結果，重みに応じた RAG 構成の選択が可能であり，全質問に対して同一の RAG 構成を適用した場合と比較して F_1 値を向上させ，実行時間を短縮した．

1　はじめに

近年，大規模言語モデル（Large Language Model, LLM）[1] は，質問応答を含む多様なタスクで高い性能を発揮している．一方で，LLM は学習データ内の知識に依存するため，未学習の知識に対して誤情報を生成するハルシネーションが生じる．また，日々更新される情報のすべてを網羅して学習することは困難である．これらの課題に対する手法として，外部知識を検索し，その結果を LLM のコンテキストとして提示する Retrieval-Augmented Generation（RAG）[2] が提案されている．しかし，従来の RAG では，外部知識に含まれるエンティティ間の関係構造を十分に活用できない．これに対し，グラフ構造をコンテキストに取り込むグラフベース RAG [3, 4] が提案されている．

質問応答の難易度は，単一の文書で解決する単純な質問から，複数文書の検索や多段階推論を要する複雑な質問まで多岐にわたる．グラフベース RAG は複雑な質問に有効であるが，検索や推論の増加により計算コストが増大する．実運用では単純な質問も多く含まれるため，常に高コストな RAG を適用すると不要なコストが生じる．一方で，複雑な質問に対して低コストな RAG を適用すると回答精度が低下する [5]．したがって，入力の特性に応じて適切な RAG 構成を選択する手法（適応的 RAG）が求められる．

既存の適応的 RAG [6, 7] には，主に三つの課題がある．第一に，LLM に基づく手法は LLM 自身が RAG 構成を調整するため計算コストが高く，複数のハイパーパラメータの総合的な調整が困難である．第二に，オンライン強化学習を用いた手法は継続的な学習を要するため，運用時の学習コストが増大する．第三に，質問の特徴に基づく手法は存在するが，文書の特徴を考慮した枠組みは議論が不足している．特に，精度指標とコスト指標を統合的に学習する手法は確立されていない．

本研究では，質問と文書の特徴を抽出して適応的に RAG 構成を選択する手法を提案する．本手法は，複数のランキングモデルと任意の重みを組み合わせることで，精度やコストに関するユーザの要求を反映する．実験の結果，重みに応じた精度とコストのトレードオフを制御できることを確認した．また，提案手法は全質問に対して同一の RAG 構成を適用した場合と比較して F_1 値を改善するとともに，実行時間を短縮した．

2　関連研究

適応的 RAG として，検索戦略やハイパーパラメータを動的に最適化する手法が提案されている．Self-RAG [6] は，LLM が検索戦略を自己評価しながら生成する手法である．同手法は生成過程での自己判断を可能にするため，LLM をファインチューニングする．これにより，モデルは生成の途中で自身の推論過程を内省し，情報検索を調整する．一方，AutoRAG-HP [7] は，RAG の複数のハイパーパ

ラメータをオンラインで最適化する手法である．同手法はハイパーパラメータ探索をオンライン多腕バンディット問題として定式化し，2 段階で探索する．上位レベルのバンディットが最適化すべきパラメータの種類を選択し，下位レベルのバンディットが具体的な値を探索することで，最適な構成を効率的に選択する．

本研究は，事前に構築したランキングモデルを用いて RAG 構成を順位付けする点で，既存手法と異なる．Self-RAG のように検索の有無を生成過程で判断せず，質問と文書の特徴に基づき順位付けを算出する．また，AutoRAG-HP と同様に複数のハイパーパラメータを同時に扱うが，オンラインの逐次試行ではなく，蓄積した評価結果に基づきランキングモデルを学習する．このオフラインで獲得した知見を新規質問に活用することで，運用時の追加探索コストや試行錯誤を不要とし，候補構成の順位付けを可能にする．さらに，精度指標とコスト指標を目的関数に組み込み，それらに対する順位付けを学習することで，多目的ランキング問題として解決する．

3 提案手法

3.1 概要

本手法は，ランキングモデル構築と推論で構成される．ランキングモデル構築では，まず質問と文書から特徴量を抽出する．次に，想定される全ての RAG 構成を用いて回答を生成する．続いて，適合率，再現率，F_1 値，および実行時間を算出する．これらの結果に基づき，各評価指標における RAG 構成の順位を予測するランキングモデルを構築する．推論では，図 1 に示すように，新規の質問と文書から抽出した特徴量を用い，ランキングモデルにより順位を予測する．図中の PR モデル，RR モデル，FR モデル，TR モデルは，それぞれ適合率，再現率，F_1 値，実行時間のランキングモデルを表す．最終的に，予測された順位とユーザが指定する重みに基づき，RAG 構成を選択する．

3.2 特徴量抽出

質問および文書から表層的特徴量と意味的特徴量を抽出する．本処理は，ランキングモデルの構築時と推論時の双方で実行する．これらの特徴量は，質問の複雑さと意図，および質問と文書の関連性を

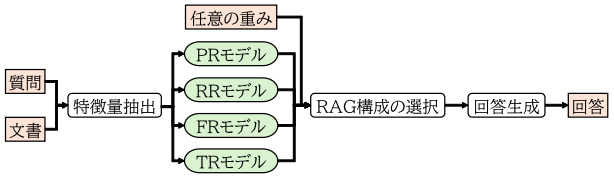

図 1: RAG 構成選択の推論プロセス

捉えるためのものである．特徴量は，ヒューリスティックな手法により選定した 7 種類を用いる．表層的特徴量は，質問に関する質問タイプ，トークン数，依存構造木の深さ，および文書に関するトークン数の 4 種類である．質問タイプは，出力が真偽判定，選択，本文抽出，文章生成のいずれであるかをルールベースで判定する．意味的特徴量は，質問文全体と文書全体の埋め込み間のコサイン類似度，質問文全体と文書から抽出したキーワードとの類似度，および質問と文書の双方から抽出した固有表現のコサイン類似度の 3 種類である．トークン数，依存構造木の深さ，固有表現の抽出には spaCy [8] を，キーワードの抽出には YAKE [9] を用いる．

3.3 データセット・ランキングモデル構築

データセット構築では，各質問に対して想定される全ての RAG 構成を適用し，生成された回答と正解を比較して BERTScore [10] に基づく適合率，再現率，F_1 値，および実行時間を算出する．実行時間は，最速値を 1 とする [0, 1] の範囲に正規化する．これら評価指標に基づき，RAG 構成間で評価値を相対比較した順位を教師ラベルとして付与する．

RAG 構成は 6 種類の項目の組み合わせとして定義する．RAG 手法には，NaiveRAG [2], LightRAG [3], および GraphRAG [4] を用いる．検索設定は，NaiveRAG では Stuff または Map_Reduce とし，LightRAG および GraphRAG ではローカル検索またはグローバル検索とする．チャンク設定は，NaiveRAG では chunk_size を 250 または 500，chunk_overlap を 50 または 100 とする．LightRAG および GraphRAG では，chunk_size を 500 または 1000, chunk_overlap を 100 または 250 とする．生成モデルには，Llama3.1:8b [11] と Qwen3:14b [12] を用い，埋め込みモデルには nomic-embed-text [13] と mxbai-embed-large [14] を用いる．

ランキングモデル構築では，第 3.2 節で定義した特徴量と各 RAG 構成を説明変数とし，4 種類の教師ラベルを目的変数として学習に用いる．

3.4 RAG 構成の選択

図 1 に示す通り，新規の質問と文書が入力されると特徴量を抽出し，ランキングモデルによって各評価指標における各 RAG 構成の順位を予測する．以下の式により総合評価値を算出し，最も高いスコアを与える構成を採用する．

$$\text{総合評価値} = \frac{W_{\text{pre}}}{P_{\text{rank}}} + \frac{W_{\text{rec}}}{R_{\text{rank}}} + \frac{W_{\text{f}_1}}{F_{1\,\text{rank}}} + \frac{W_{\text{time}}}{C_{\text{time\_rank}}}$$

ただし，重みは以下の制約を設ける．

$$W_{\text{pre}} + W_{\text{rec}} + W_{\text{f}_1} + W_{\text{time}} = 1$$

ここで，P_{rank}，R_{rank}，$F_{1\,\text{rank}}$ および $C_{\text{time\_rank}}$ はそれぞれ適合率，再現率，F_1 値，実行時間の予測値に基づく順位を表す．各項において順位の逆数を用いることで，より上位の性能を持つ手法に対し，高いスコアを与える設計とする．最終的に，最大の総合評価値を得た RAG 構成を回答生成に用いる．

4 実験

4.1 概要

本実験の目的は，QA タスクにおいて RAG 構成を適応的に選択するランキングモデルの有効性を評価することである．実験は，(i) ランキングモデルの性能評価，および (ii) 選択された RAG 構成を用いた RAG の性能評価の二つの観点から構成した．

4.2 評価用データセット

本実験では，SQuAD [15]，HotpotQA [16]，2WikiMultihopQA [17]，MuSiQue [18] の四つの QA データセットを用いた．各データセットは，質問，対応する文書，および正解から構成される．多様な難易度で評価するため，四つのデータセットを統合して用いた．各データセットから，訓練用として 100 事例，評価用として 20 事例をランダムに抽出した．

4.3 ランキングモデル

ランキング学習用モデルとして，LightGBM [19] を採用した．本モデルは勾配ブースティング決定木に基づく手法であり，非線形な特徴間の相互作用を高精度に学習できる．XGBoost と CatBoost と比較した予備実験において最も高い性能を示したため，本手法の採用に至った．

4.4 提案手法の設定

本研究の実装詳細および評価における重み付けの設定について述べる．NaiveRAG の実装には LangChain [20] を，グラフベース RAG には GitHub [21, 22] で公開されている実装を用いた．生成モデルと埋め込みモデルについては，Ollama [23] を用いて実装した．GPU には NVIDIA A100 を用い，提案手法で選択しないハイパーパラメータはデフォルト値に固定した．また，すべての実験で乱数シードを固定し，再現性を確保した．評価項目 (ii) においては，目的関数の重み付けが異なる五つの設定を用意した．第一の設定は精度とコストのバランスを重視し，$W_{\text{f}_1} = W_{\text{time}} = 0.5$ とした．第二の設定は適合率を重視し，$W_{\text{pre}} = 1.0$，第三の設定は再現率を重視し，$W_{\text{rec}} = 1.0$ とした．第四の設定は F_1 値重視で，$W_{\text{f}_1} = 1.0$ を重視し，第五の設定は実行時間を重視し，$W_{\text{time}} = 1.0$ とした．

4.5 ベースライン

本実験では，比較対象として複数のベースラインを設定した．評価項目 (i) では，ランダムに生成したランキングをベースラインとし，異なるシード値で 10 回試行した平均値を算出した．評価項目 (ii) では，五つのベースラインを設定した．第一に，低コスト構成として，NaiveRAG の（stuff, chunk_size=250, chunk_overlap=50, llama3.1:8b, nomic-embed-text）を全質問に適用する「標準設定」を設けた．第二に，提案手法（F_1 値・時間）設定で頻出した構成として，NaiveRAG の（stuff，chunk_size=500, chunk_overlap=50，llama3.1:8b，nomic-embed-text）を固定した「最頻出設定」を設けた．第三に，候補から RAG 構成を無作為に選ぶ「ランダム選択」を設けた．最後に，候補集合から評価指標が最大となる設定を事後的に選択する「オラクル」を定義し，理論上の上限とした．

4.6 評価指標

ランキングモデルの評価指標として NDCG@k および Hit@k を用いる．これらの指標は，ランキング上位に正解となる候補が現れることを重視しており，推論プロセスにおける実用的な性能を評価できる．また，RAG の回答性能を評価するため，BERTScore に基づく適合率，再現率，F_1値，および実行時間を採用する．これらの指標を用い，ランキ

表 1: ランキング予測の評価結果（%）

目的変数	手法	NDCG@1	NDCG@5	Hit@1	Hit@5
適合率	LightGBM	**67.15**	**68.48**	**60.00**	**70.00**
	ランダム	22.29	23.05	16.13	39.88
再現率	LightGBM	**71.46**	**72.07**	**48.75**	**60.00**
	ランダム	33.47	34.37	8.63	32.00
F_1 値	LightGBM	**68.52**	**67.81**	**56.25**	**65.00**
	ランダム	22.67	23.32	10.75	36.13
時間	LightGBM	**99.71**	**99.54**	**81.25**	**100.00**
	ランダム	50.56	51.75	1.88	10.50

ングモデルと重みの有効性を検証する．

4.7 結果

表 1 にランキング予測の評価結果を示す．太字は一番高い値，下線は次に高い値を示す．LightGBM は，すべての予測においてランダムベースラインを一貫して上回った．NDCG@1 は，適合率・再現率・F_1 値の予測で 67.15〜71.46%，実行時間の予測で 99.71% を示した．また，Hit は k の増加に伴い上昇しており，最良候補を上位 k 件に含めるという観点で安定した性能を示した．

表 2 に，ランキング上位 k 件（@1，@5）から RAG 構成を選択する場合の性能を，五つの重み設定および各ベースラインと比較した結果を示す．これらの結果から，提案手法の有効性が四つの観点で示された．第一に，重み設定によって精度指標と実行時間のトレードオフを同一枠組みで制御できる点である．第二に，標準設定と比較して F_1 値を改善しつつ，実行時間を短縮できる点である．特に，提案手法 @1（F_1 値・時間）は 評価用データセットの平均値において標準設定と比較し，F_1 値を 3.08 %向上させ，実行時間を 3.75 秒短縮した．第三に，ランダム選択のような過度な高コストを回避し，運用可能なコスト範囲に収められる点である．第四に，時間最小化の観点ではオラクルに近い性能を示し，低コスト運用における実用性が高い点である．一方で，精度最大化を目的としたオラクルとの間には依然として差が残る結果となった．

4.8 考察

実験結果は，提案手法が平均性能を向上させるだけでなく，評価指標の重み付けに応じて，重視すべき対象を切り替える「適応的 RAG」として機能することを示唆している．また，単一の固定構成で全ての質問に対応する手法は，精度とコストのトレードオフにおいても非効率的であることが明らかになった．さらに，ランダム選択が実行時間の増加と精度低下を招いたことから，候補空間が広大な場合には選択機構の導入が不可欠であると結論付けられる．

表 2: RAG 構成選択の評価結果

手法	適合率（%）	再現率（%）	F_1 値（%）	時間（秒）
提案手法 @1				
F_1 値・時間	41.07	55.02	43.58	1.43
適合率重視	39.16	52.03	41.55	5.49
再現率重視	**43.85**	**57.05**	**46.25**	9.88
F_1 値重視	42.12	56.44	45.08	7.24
時間重視	40.94	55.02	43.55	**1.41**
提案手法 @5				
F_1 値・時間	40.18	54.64	43.11	4.83
適合率重視	39.01	53.20	41.86	7.10
再現率重視	42.37	56.36	45.33	8.79
F_1 値重視	40.63	55.69	43.67	8.99
時間重視	30.57	46.10	33.57	2.16
ベースライン				
標準設定	38.61	51.76	40.50	5.18
最頻出設定	40.94	55.02	43.55	**1.41**
ランダム選択	0.06	23.37	8.20	51.29
オラクル				
適合率最良	55.08	68.17	58.05	14.91
再現率最良	51.36	72.46	58.04	14.91
F_1 値最良	53.83	70.38	59.72	15.90
時間最良	40.86	54.99	43.49	1.27

5 おわりに

本研究では，質問に応じて RAG 構成を適応的に選択し，精度とコストのトレードオフを制御する枠組みを提案した．実験ではランキング予測の妥当性を評価し，ランキングモデルがランダム選択を上回ることを確認した．実行時間を目的とするランキング予測は良好な精度を示し，上位候補の提示において安定した性能を達成した．重み設定に基づく性能比較では，重みの変化が精度と実行時間のトレードオフに影響を与えることを確認した．これにより，運用目標に応じた調整が可能である．全質問に対して同一の RAG 構成を適用した場合と比較し，提案手法は F_1 値を向上させつつ実行時間を短縮した．単一の固定構成では質問の多様性への対応が困難であることを定量的に示した．今後の課題は，ランキングモデルの精度向上，ユーザ意図から重み設定の自動化，および評価指標の拡張である．

謝辞

本研究成果の一部は，国立研究開発法人新エネルギー・産業 技術総合開発機構 (NEDO) の委託業務 (JPNP20006, JPNP25006) の結果得られたものです．本研究は JSPS 科研費 23K11221，25K03232 の助成を受けたものです．

参考文献

[1] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2025.

[2] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In **Proceedings of the 34th International Conference on Neural Information Processing Systems**, NeurIPS 2020, Red Hook, NY, USA, 2020. Curran Associates Inc.

[3] Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. LightRAG: Simple and fast retrieval-augmented generation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, **Findings of the Association for Computational Linguistics: EMNLP 2025**, pp. 10746–10761, Suzhou, China, November 2025. Association for Computational Linguistics.

[4] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization, 2025.

[5] Xiaqiang Tang, Jian Li, Nan Du, and Sihong Xie. Adapting to non-stationary environments: Multi-armed bandit enhanced retrieval-augmented generation on knowledge graphs. **Proceedings of the AAAI Conference on Artificial Intelligence**, Vol. 39, No. 12, pp. 12658–12666, Apr. 2025.

[6] 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.

[7] Jia Fu, Xiaoting Qin, Fangkai Yang, Lu Wang, Jue Zhang, Qingwei Lin, Yubo Chen, Dongmei Zhang, Saravan Rajmohan, and Qi Zhang. AutoRAG-HP: Automatic online hyper-parameter tuning for retrieval-augmented generation. In **Findings of the Association for Computational Linguistics: EMNLP 2024**, pp. 3875–3891, Miami, Florida, USA, 2024. Association for Computational Linguistics.

[8] Explosion. spacy: Industrial-strength natural language processing in python. https://spacy.io/, 2016.

[9] INESC TEC. Yake! (yet another keyword extractor). http://yake.inesctec.pt/, 2018.

[10] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with bert. In **International Conference on Learning Representations**, 2020.

[11] Meta. The llama 3 herd of models, 2024.

[12] QwenTeam. Qwen3 technical report, 2025.

[13] Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic embed: Training a reproducible long context text embedder, 2025.

[14] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. Open source strikes bread — new fluffy embedding model. https://www.mixedbread.com/blog/mxbai-embed-large-v1, 2024.

[15] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Jian Su, Kevin Duh, and Xavier Carreras, editors, **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[16] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**, pp. 2369–2380, Brussels, Belgium, 2018. Association for Computational Linguistics.

[17] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In **Proceedings of the 28th International Conference on Computational Linguistics**, pp. 6609–6625, Barcelona, Spain (Online), 2020. International Committee on Computational Linguistics.

[18] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. **Transactions of the Association for Computational Linguistics**, Vol. 10, pp. 539–554, 2022.

[19] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, **Advances in Neural Information Processing Systems**, Vol. 30. Curran Associates, Inc., 2017.

[20] LangChain Team. LangChain. https://www.langchain.com/, 2023.

[21] HKUDS. LightRAG: Simple and fast retrieval-augmented generation. https://github.com/HKUDS/LightRAG, 2024.

[22] Microsoft. GraphRAG. https://github.com/microsoft/graphrag, 2024.

[23] Ollama. Ollama. https://ollama.com/g, 2025.

A　データセットの基本統計量

本評価実験では，SQuAD，HotpotQA，2WikiMultihopQA，および MuSiQue の四つのデータセットを統合した．表 3 には，質問と文書から抽出した特徴量の基本統計量を示す．表 4 には，算出された評価値の基本統計量を示す．

表 3: 質問・文書の特徴量の基本統計量

		質問			文書	
データ	指標	トークン数	固有表現数	依存構造木の深さ	トークン数	固有表現数
訓練	平均値	16.30	1.68	6.22	656.78	88.98
	標準偏差	9.24	1.33	2.26	616.00	84.93
	中央値	15.00	1.00	6.00	546.50	81.00
	最小値	5.00	0.00	2.00	2.00	0.00
	最大値	117.00	9.00	18.00	3046.00	585.00
テスト	平均値	14.34	1.33	5.84	655.92	85.96
	標準偏差	4.56	0.93	1.94	637.72	83.41
	中央値	13.50	1.00	5.00	411.00	68.50
	最小値	6.00	0.00	3.00	14.00	1.00
	最大値	27.00	4.00	11.00	3277.00	334.00

表 4: 適合率・再現率・F_1 値・実行時間の基本統計量

データ	指標	適合率	再現率	F_1 値	実行時間（秒）	正規化実行時間
訓練	平均値	0.065	0.241	0.089	50.949	0.974
	標準偏差	0.207	0.338	0.216	86.842	0.045
	中央値	0.000	0.042	0.000	30.798	0.984
	最小値	0.000	0.000	0.000	0.354	0.000
	最大値	1.000	1.000	1.000	1926.923	1.000
テスト	平均値	0.072	0.315	0.098	68.224	0.970
	標準偏差	0.226	0.398	0.230	93.297	0.041
	中央値	0.000	0.082	0.000	37.385	0.984
	最小値	0.000	0.000	0.000	0.338	0.000
	最大値	1.000	1.000	1.000	2250.575	1.000

B　ケーススタディ

質問「Where did the director of film Nalini By Day, Nancy By Night graduate from?」は，特定人物（監督）の卒業大学を問うものである．再頻出設定では「temple university」が得られ，提案手法の @1 でも同様に「temple university」を出力した．一方で，提案手法の @5 には異なる設定が含まれ，その実行結果として「mount holyoke college」のような正解となる回答が生成された．これは，上位候補を広げるほど探索空間は拡大することで，提案手法は候補集合として固定ハイパーパラメータでは得ることのできない，別回答を提示可能であることを示した．

C　実用化に向けた評価

本章では，提案手法が出力する「上位 K 個の RAG 構成」を，実運用に近い状況での利用を想定し，その有効性を検証した．具体的には，ユーザが上位 K 個の RAG 構成をそれぞれについて RAG を実行し，得られた性能指標に基づいて最終的に一つの RAG 構成を採用するという運用フローを想定する．実験手順として，各質問に対して上位 K 個の候補を用いて推論し，それぞれの出力を正解データと比較して適合率，再現率，F_1 値，および実行時間を算出する．その後，運用上の意思決定基準を模した「重視方針」に従い，K 個の結果の中から一つの候補を選択する．選択された候補に対応する指標値をその質問の代表値とし，全質問にわたる平均を計算することで，方針別の性能を算出する．具体的には，「F_1 値重視」では，各質問について K 個の RAG 構成のうち F_1 値が最大となる RAG 構成を一つ採用する．同様に，「適合率重視」および「再現率重視」では，それぞれ対応する指標が最大となる RAG 構成を一つ選択する．「時間重視」では，実行時間が最小となる RAG 構成を一つ選択し，速度を重視した運用を表現する．

表 5 に示す結果より，K を増やすことで目的指標を基準とした選択の自由度が高まり，性能が改善する傾向が確認された．特に精度系の方針において，$K = 5$（@5）の結果は $K = 3$（@3）を一貫して上回った．例えば，再現率重視では，@3 の F_1 値 48.68%に対し，@5 では 50.53%まで上昇し，再現率も 59.37%から 60.93%へと改善した．これは，ランキング上位集合の中に「目的指標の観点でより良い設定」が含まれている確率が K とともに高まり，最終選択でそれを特定できたことを示している．すなわち，提案手法は順位の厳密性に加え，ランキング上位集合の質により実運用での有効性を発揮すると言える．一方，K の増加が常に実行時間の増大を招くわけではない点も重要である．再現率重視の結果を見ると，@3 の実行時間 9.88 秒に対し，@5 では 7.13 秒と短縮している．ただし，選定プロセスにおいて K 回の試行を要するため，システム全体の総処理時間は増加することに留意が必要である．

表 5: RAG ハイパーパラメータ選択の評価結果

手法	適合率（%）	再現率（%）	F_1 値（%）	実行時間（秒）
提案手法 @3				
適合率重視	44.47	56.67	46.86	6.20
再現率重視	**44.70**	59.37	**48.68**	9.88
F_1 値重視	44.46	**59.58**	47.68	7.78
時間重視	41.17	55.18	43.75	**1.27**
提案手法 @5				
適合率重視	45.24	58.42	48.03	6.82
再現率重視	**46.41**	60.93	**50.53**	7.13
F_1 値重視	45.40	**62.19**	50.25	7.76
時間重視	40.86	54.99	43.49	**1.27**