R^2AG: A Robust Retrieval-Augmented Generation Benchmark for Query Entry Errors

Anonymous ACL submission

Abstract

Retriever-augmented generation (RAG) has become a widely adopted approach for enhancing the factual accuracy of large language models (LLMs). While current benchmarks evaluate the performance of RAG methods from various perspectives, they share a common assumption that user queries used for retrieval are error-free. However, in real-world interactions between users and LLMs, query entry errors such as keyboard proximity errors, visual similarity errors, and spelling errors are frequent. The impact of these errors on subsequent LLM outputs, as well as the robustness of current RAG methods against such errors, remains largely unexplored. To bridge this gap, we propose R^2AG, the first robust RAG benchmark designed specifically to evaluate performance against query entry errors. We collect six widely used datasets and inject the three types of query entry errors into randomly selected user queries at proportions of 40% and 80%. Extensive in-domain and cross-domain experiments reveal that: (1) state-of-the-art RAG retrievers and query rewriting methods show poor robustness to query entry errors, with average token F1 scores of approximately 27% and 11% across the datasets; (2) contrastive learning with correct and corrupted query pairs can improve retriever robustness, but reduces RAG performance for correct queries; (3) our adaptive retrieval selector that uses different retrievers for correct and corrupted queries mitigates the issues introduced by contrastive learning.

1 Introduction

Retriever-augmented generation (RAG) (Borgeaud et al., 2022; Lewis et al., 2020; Chen et al., 2024), which integrates retrieval mechanisms to incorporate external knowledge into large language models (LLMs), has become a widely adopted approach. By retrieving knowledge from external sources, RAG addresses issues such as insufficient knowledge and hallucinations in LLMs (Tonmoy et al., 2024; Gao et al., 2023b), thereby improving the accuracy and fidelity of their responses.

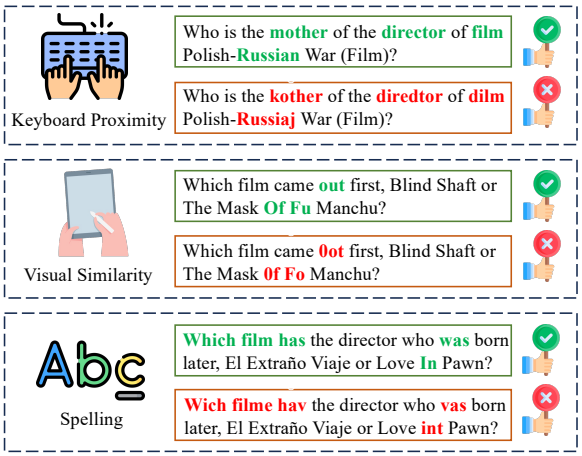


Figure 1: Examples of three types of query entry errors.

Current RAG benchmarks evaluate the performance of RAG methods from various perspectives, providing valuable insights into their effectiveness in different scenarios. For instance, they assess the fidelity of LLM-generated content (Es et al., 2024), the ability to refuse to answer (Chen et al., 2024), and the capacity to handle counterfactual information (Liu et al., 2023). However, they share a common assumption that user queries used for retrieval are error-free. In real-world settings, as illustrated in Figure 1, user queries often contain entry errors including keyboard proximity errors, visual similarity errors, and spelling errors. The impact of these errors on subsequent LLM outputs, as well as the robustness of current RAG methods against such errors, remains largely unexplored.

To fill this gap, we propose a new RAG benchmark dataset, namely R^2AG, for evaluating the Robustness of RAG against query entry errors. Specifically, we collect four direct QA datasets (TriviaQA (Joshi et al., 2017), Natural Questions (Kwiatkowski et al., 2019), PopQA (Mallen et al., 2022), WebQuestions (Berant et al., 2013))

and two multi-hop QA datasets (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020)). Using the nlpaug (Ma, 2019) tool, we inject three types of query entry errors: spelling errors, keyboard proximity errors, and visual similarity errors, reflecting real-world error distribution patterns with a 3:1:1 ratio. For each query, we apply a 30% probability of selecting a word, and for each selected word, a 30% probability of corrupting a character. This probability mimics typical user query behaviors, ensuring a realistic test environment for RAG models. As these errors do not affect the underlying user information need, we retain the original RAG labels for the corrupted queries. To assess model robustness under varying levels of noise, we generate two versions of the R^2AG by corrupting 40% and 80% of the queries, representing moderate and high-error scenarios.

We further evaluate the in-domain and cross-domain performance of RAG (i.e., training on HotpotQA and testing on either the same or other datasets) using three potential solutions to address query entry errors *First*, we select the current state-of-the-art retrievers used in RAG, namely BGE (Xiao et al., 2023) and GTE (Li et al., 2023), from the MTEB leaderboard (Muennighoff et al., 2022), to assess their robustness against query entry errors. *Second*, we leverage two large language models, Qwen2 (Yang et al., 2024) and LLama3 (AI@Meta, 2024), to correct potentially corrupted input queries before testing RAG performance with the BGE and GTE retriever. *Third*, inspired by query reformulation research (Gao et al., 2023b) in information retrieval, we employ HyDE (Gao et al., 2023a), a specialized query reformulation model, to rewrite input queries, which are then processed through the BGE-based RAG framework. Experiments reveal that these methods exhibit poor robustness to query entry errors, especially for HyDE.

To address this issue, we propose a contrastive learning (Gao et al., 2021)-based approach that compares pairs of corrupted queries and relevant documents, as well as an adaptive retrieval selector that employs different retrievers for correct and corrupted queries. Experiments demonstrate that the former approach improves retriever robustness but compromises RAG performance on correct queries, whereas the latter further improves the robustness.

To summarize, our contributions are as follows:

- To the best of our knowledge, we are the first to investigate robustness against query entry errors in RAG research, focusing on three representative error types: keyboard proximity, visual similarity, and spelling.
- We construct a benchmark dataset, R^2AG, based on six widely-used RAG datasets, incorporating two levels of noise through the explicit injection of three types of errors.
- Extensive experiments conducted on R^2AG demonstrate that state-of-the-art RAG retrievers, large language models, and query reformulation methods show poor robustness to query entry errors.
- We propose two solutions to improve robustness against query entry errors: (1) contrastive learning, which enhances retriever robustness but reduces RAG performance on correct queries; (2) an adaptive retrieval selector, which applies different retrievers for correct and corrupted queries, resulting in further improvements on robustness.

2 Related Work

2.1 RAG in LLM Era

In the era of Language Models (LLMs), the way of obtaining information has changed, with more and more users preferring to obtain information through LLM rather than search engines (Ai et al., 2023; Dai et al., 2024). By combining information retrieval and generation, the emergence of RAG technology allows LLM to gain new knowledge from external databases as a supplement, making its generated content more accurate and reliable (Lewis et al., 2020; Gao et al., 2023b). Widely used RAG datasets (Yang et al., 2018; Kwiatkowski et al., 2019; Joshi et al., 2017) have been manually annotated and reviewed, without incorporating query entry errors into the dataset. Existing RAG benchmarks primarily assess the quality of content generated by LLMs or the LLM's ability to process external information. RAGAS (Es et al., 2024) and ARES (Saad-Falcon et al., 2024) evaluate the contextual relevance and fidelity of LLM-generated content. RGB (Chen et al., 2024) tests the robustness of LLM against noisy documents and the ability to refuse to answer, while RECALL (Liu et al., 2023) analyzes the LLM's processing capability regarding counterfactual information. However, they all assume that the queries used for retrieval are

Type	Dataset	Source	#Query	Avg. #Chars/Query			Avg. #Words/Query		
				0% Prob	40% Prob	80% Prob	0% Prob	40% Prob	80% Prob
QA	NQ	Wiki	3610	48.4	48.7	48.9	9.4	9.4	9.5
	PopQA	Wiki	14267	37.1	37.7	38.3	6.7	6.9	6.9
	TrivalQA	Wiki & Web	11313	69.1	69.6	70.1	12.6	12.7	12.8
	WebQA	Google Freebase	2032	38.0	38.1	38.3	6.8	6.9	7.0
Multi-Hop QA	HotpotQA	Wiki	7405	94.5	95.1	95.6	16.4	16.5	16.6
	2wiki	Wiki	12576	68.1	68.8	69.4	12.4	12.5	12.6

Table 1: The statistics of six datasets used in R^2AG. "Source" refers to the knowledge source of each dataset. "#Query" denotes the number of queries. "0% Prob", "40% Prob", "80% Prob" represent the proportions of corrupted queries in the dataset at 0%, 40%, and 80%, respectively. "Avg. #Char/Query" indicates the average number of characters per query. "Avg. #Words/Query" refers to the average number of words per query.

correct, without considering the actual scenarios where users may enter corrupted queries. In the increasingly popular era of LLM, this cannot well evaluate the real capabilities of RAG technology. Therefore, this paper focuses on establishing an RAG evaluation framework that includes corrupted queries, which can help evaluate the robustness of RAG models and promote the further development of RAG technology in the era of LLM.

2.2 Query rewriting

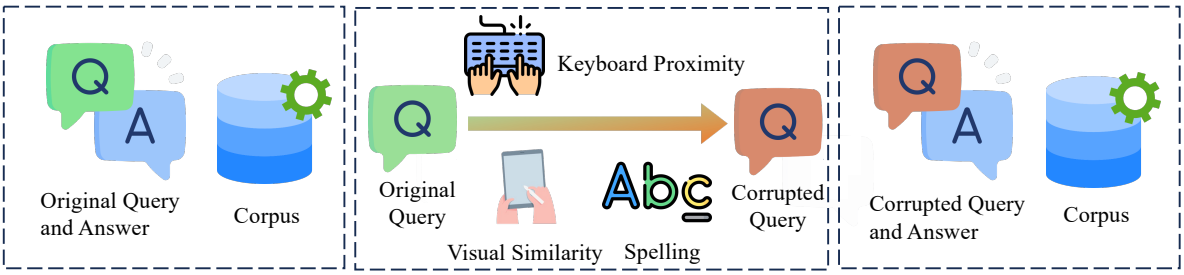
Query rewriting technology considers that the original user query is not always the optimal query that can help LLM in retrieval. In addition to letting LLM directly rewrite the query, HyDE (Gao et al., 2023a) allows LLM to generate a hypothetical document based on the query to answer the query, and then complete the retrieval through the generated hypothetical document. It believes that the generated hypothetical document may be better aligned with the retrieved document. Query2doc (Wang et al., 2023) concatenates the pseudo document generated by LLM and the original query as a new query. Rewrite-retrieve-read (Ma et al., 2023) proposed rewriting by fine-tuning a rewriter. BEQUE (Ye et al., 2023) improves the recall effect of long-tail queries through fine-tuning and reinforcement learning methods. The above query rewriting methods do not consider that the query itself is corrupted, thus ignoring that query rewriting may accumulate and amplify errors, which will seriously affect the final RAG performance.

3 R^2AG Dataset Construction

We focus on RAG in this study, which is formulated as follows: given a query $q \in \mathcal{Q}$ (where $\mathcal{Q}$ is the set of all possible queries) and an external knowledge base $K = \{d_1, d_2, \ldots, d_N\}$ consisting of N documents, the goal of RAG is to generate a response $a \in \mathcal{A}$ (where $\mathcal{A}$ is the set of possible answers) by leveraging both retrieval from the knowledge base and generation from a LLM. Unlike previous datasets, which assume that q is error-free, we consider a more practical scenario in which q may be corrupted by three types of query entry errors. As illustrated in Figure 2, our R^2AG dataset is constructed through the following steps.

Step1: Selection of RAG Dataset. Following FlashRAG (Jin et al., 2024), we collect and extend six widely-used RAG datasets to form our **R^2AG**, which includes four direct QA datasets (TriviaQA (Joshi et al., 2017), Natural Questions (Kwiatkowski et al., 2019), PopQA (Mallen et al., 2022), WebQuestions (Berant et al., 2013)) and two multi-hop QA datasets (HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020)). Each dataset follows the format "*question, gold answer*", representing the user query q and the gold answer a, respectively. The corpus K used for retrieval, also referred to as the external knowledge base, is set to the Wikipedia corpus. Please note that to comprehensively evaluate the robustness of existing methods against query entry errors, we conduct both in-domain and cross-domain robustness assessments. Following (Xu et al., 2024), we use HotpotQA as the source dataset, meaning we fine-tune the retrieval model exclusively on HotpotQA. Testing on HotpotQA constitutes in-domain evaluation, while testing on other datasets represents cross-domain evaluation.

Step 2: Query Corruption. We utilize the nlpaug tool (Ma, 2019) to inject three types of query entry errors into the six collected datasets, forming the corrupted queries: (1) **Keyboard Proximity Errors.** When users interact with LLMs via a keyboard, mistyping may occur as a result of pressing adjacent keys. To simulate this, we replace correct letters with nearby letters on the keyboard. (2) **Vi-**



(a) Selection of RAG Dataset **(b) Query Corruption** **(c) Label Matching**

Figure 2: The construction process of R^2AG datasets. (a) Selection of RAG Dataset. (b) Query corruption through three scenarios: keyboard proximity errors, visual similarity errors and spelling errors. (c) Label matching.

sual Similarity Errors. When users input words through handwriting, recognition tools may misinterpret characters due to irregular handwriting or inaccurate OCR algorithms, resulting in morphological errors. To simulate these handwriting input errors, we replace correct letters with visually similar ones. (3) **Spelling Errors.** Users may occasionally forget the correct spelling of a word and input an approximation, leading to spelling errors in the query. We simulate these errors by replacing words using a spelling error dictionary. Specifically, we apply a 30% probability of selecting a word in each query, and for each selected word, a 30% probability of corrupting a character. These probabilities reflect typical user behavior, creating a realistic test environment for RAG models.

Step 3: Label Matching. Since we set relatively low probabilities for both selecting a word and corrupting a character, we assume the corruption does not affect the underlying user information need and realistically simulates typical user query behavior. Therefore, we retain the original RAG labels for the corrupted queries. In other words, for an original data sample (q, a), we replace it with (q', a) where q' is the corrupted version of q containing one of the three entry errors, while a remains unchanged. Additionally, to evaluate model robustness under different levels of noise, we generate two versions of the R^2AG dataset by corrupting 40% and 80% of the queries, representing moderate and high-error scenarios.

Dataset Statistics and Analysis. Table 1 presents the statistical analysis of the six datasets we constructed. It can be observed that the difference in the average number of words per query between corrupted queries (with error ratios of 40% and 80%) and original queries is not significant. This similarity indicates that our corruption strategy effectively mirrors real-world scenarios of user query entry errors. Additionally, our corruption strategy does not alter the syntactic structure of the sentences, as shown by the minimal difference in the average query length between original and corrupted queries in Table 1, further ensuring the quality of our R^2AG dataset.

Evaluation. Following (Jin et al., 2024), we use EM (Exact Match), F_1 (token-level F_1 score), and Acc (Accuracy) to evaluate the effectiveness and robustness against query entry errors of RAG methods. Additionally, we have developed a Python framework that facilitates the easy reproduction of experiments and the integration of new datasets and additional RAG methods. Further details on the datasets and evaluation code can be found at `https://anonymous.4open.science/r/R2AG-CB55`.

4 Methodology

In this section, we present the contrastive learning-based approach and the adaptive retrieval model selection method, both designed to enhance robustness against query entry errors.

4.1 Contrastive Learning-Based Method

Contrastive learning (CL) is a self-supervised learning technique designed to learn robust representations by contrasting positive and negative examples. Its advantage lies in enhancing the model's discriminative power by bringing semantically similar pairs closer in the embedding space while pushing dissimilar pairs further apart. This makes it particularly effective in scenarios where the model needs to distinguish subtle differences between inputs, such as query entry errors. Thus, we leverage CL to train the model to recognize and retrieve relevant documents even when queries are corrupted.

Specifically, we use the HotpotQA dataset, introducing an 80% corrupted query ratio to construct

contrastive pairs in the format (q, a) and (q', a), where q and q' respectively denotes the original and corrupted query, and a denotes the golden LLM response. We then fine-tuned BGE (Xiao et al., 2023) and GTE (Li et al., 2023) models using contrastive learning on this dataset, with positive examples being the relevant documents corresponding to the original queries in HotpotQA. For negative example sampling, we included a hard negative example for each corrupted query, randomly chosen from the original HotpotQA corpus, along with randomly selected in-batch soft negative examples. The training objective is:

$$\mathcal{L} = -\log \frac{e^{\mathrm{sim}(\mathbf{q_i'}, \mathbf{d_i^+})/\tau}}{e^{\mathrm{sim}(\mathbf{q_i'}, \mathbf{d_i^+})/\tau} + \sum_{j=1}^{N} e^{\mathrm{sim}(\mathbf{q_i'}, \mathbf{d_j^-})/\tau}}, \tag{1}$$

where $\mathbf{q_i'}$, $\mathbf{d_i^+}$, and $\mathbf{d_i^-}$ denote the embeddings of the i-th corrupted query, the positive example, and the negative example, respectively. The function $\mathrm{sim}(\cdot)$ represents the cosine similarity function, N is the batch size, and τ is the temperature.

4.2 Adaptive Retrieval Model Selection

The other solution for improving robustness involves adding an extra selector to the RAG framework to determine whether the current query is corrupted. If the query is correct, we use the retrieval model trained on the original data; if the query is corrupted, we switch to the retrieval model trained on corrupted data using contrastive learning (Sec. 4.1). We refer to this approach as the adaptive retrieval model selection method (ASR).

Specifically, we construct a query classification dataset based on HotpotQA, maintaining a 1:1 ratio of correct to corrupted queries. The corresponding validation set of HotpotQA is used to create a validation set for query classification. We then train a BERT-based selector (Devlin et al., 2019) f_{ARS} as follows:

$$p_c = P(\mathrm{correct}|q) = f_{\mathrm{ARS}}(x; \theta_{\mathrm{ARS}}), \tag{2}$$

where θ_{ARS} represents the parameters of f_{ARS}, $P(\mathrm{correct}|q)$ represents the probability that the query q is correct. If $P(\mathrm{correct}|q)$ exceeds a certain threshold (set to 0.5), the query is considered correct.

5 Experiments

Using our R^2AG dataset, we aim to: (1) assess the robustness of state-of-the-art RAG retrieval models, large language models, and query reformulation methods against query entry errors; and (2) evaluate the effectiveness of the two proposed solutions—contrastive learning (CL)-based and adaptive retrieval model selection (ARS)—in improving robustness against these errors.

5.1 Experimental Settings

5.1.1 Datasets and Metrics

In the main experiment, we selected our modified RAG dataset to conduct experiments on RAG tasks. Specifically, we chose four QA datasets: TriviaQA (Joshi et al., 2017), Natural Questions (NQ) (Kwiatkowski et al., 2019), PopQA (Mallen et al., 2022), WebQuestions (WebQA) (Berant et al., 2013), and two Multi-Hop QA datasets: HotpotQA (Yang et al., 2018) and 2WikiMultihopQA(2wiki) (Ho et al., 2020) for our experiments. Following (Jin et al., 2024), we used the Wikipedia data from December 2018 as the retrieval corpus. Regarding the evaluation metrics, following (Jin et al., 2024), we selected the widely used token-level F_1 score as our evaluation metric.

5.1.2 Retrieval and Generation Models

In our main experiment, we selected the sentence embedding models with SOTA performance on the MTEB leaderboard (Muennighoff et al., 2022), namely bge-base-en-v1.5 (Xiao et al., 2023) and gte-base-en-v1.5 (Li et al., 2023), as the retrieval models, denoted as **BGE-Base** and **GTE-Base**. Other retrieval models can also be adapted to our benchmark. As Sec. 4.1 described, we trained them using standard contrastive learning on the HotpotQA (Yang et al., 2018) retrieval dataset we constructed, denoted as **BGE-CL** and **GTE-CL**.

For the generation models, we chose the latest Llama3-8B-Instruct (AI@Meta, 2024) and Qwen2-7B-Instruct (Yang et al., 2024) as the main experimental generation models. They have been proven to have strong performance in multiple RAG tasks. We also used the LLM zero-shot method as a comparison, denoted as **Naive Generation**.

5.1.3 Query Rewriting Methods

For the query rewriting baseline, considering the computational cost, we chose methods that do not require training. Specifically, we chose **Correct**: Use LLM to directly correct the query, and then use the corrected query as the new query for retrieval. **HyDE** (Gao et al., 2023a): Use LLMs to generate a pseudo-document based on the original

Dataset	HotpotQA	NQ	PopQA	TrivalQA	WebQA	2wiki	HotpotQA	NQ	PopQA	TrivalQA	WebQA	2wiki	Overall
Method	**Llama3**						**Qwen2**						
	80% Corrupted Queries												
Naive Generation	18.00	19.03	13.32	41.51	**26.64**	**17.86**	19.78	15.60	13.03	35.84	24.60	**24.89**	22.51
BGE-Base	24.36	25.03	24.53	41.88	21.64	14.56	28.66	26.63	27.40	43.03	25.10	23.36	27.18
BGE-CL	**26.62**	27.19	**27.26**	**44.59**	22.85	15.33	**31.39**	**27.79**	**30.25**	**45.93**	**25.94**	24.17	**29.11**
Correct	22.48	**28.17**	25.43	38.00	24.27	14.27	25.46	21.96	25.37	35.46	23.16	22.95	25.58
HyDE	6.06	15.95	2.34	19.95	11.33	4.11	8.88	12.54	6.65	22.37	12.38	5.02	10.63
	40% corrupted Queries												
Naive Generation	23.11	24.37	15.68	49.29	**31.94**	**23.63**	24.21	21.67	15.30	42.28	28.20	27.63	27.27
BGE-Base	30.01	**34.12**	33.80	52.23	27.65	17.04	**35.84**	**35.99**	35.98	53.54	30.65	29.30	34.67
BGE-CL	**31.10**	33.61	**35.17**	**52.81**	29.52	18.19	35.45	35.91	**37.40**	**53.84**	**31.87**	**30.39**	**35.44**
Correct	23.15	32.45	31.15	35.62	24.89	14.82	25.47	24.07	29.01	32.99	22.87	24.33	26.73
HyDE	6.58	16.92	2.04	19.74	12.21	4.41	10.13	15.16	7.92	24.82	13.27	5.67	11.57
	0% corrupted Queries												
Naive Generation	26.32	30.55	19.95	56.68	**37.98**	**29.00**	26.40	24.99	18.39	47.69	33.84	34.09	32.16
BGE-Base	**37.65**	**42.34**	40.46	**62.67**	34.55	20.53	**43.10**	**44.98**	43.41	**64.15**	36.69	**35.63**	**42.18**
BGE-CL	36.46	40.24	**41.10**	61.60	35.70	20.07	41.82	41.60	**44.70**	62.46	**37.36**	33.44	41.38
Correct	22.09	34.78	37.11	33.52	27.28	16.28	26.39	23.53	31.52	32.33	22.49	28.47	27.98
HyDE	7.42	18.79	2.02	21.71	11.49	4.73	10.41	16.49	7.70	27.74	16.27	4.82	12.47

Table 2: The overall performance of the RAG task under six datasets and three different error proportions of queries scenarios when the retrieval model is BGE, and the generator models are Llama3 and Qwen2. The optimal results are presented in bold. The **"overall"** column represents the average result of that row, which is the average result of the method across all datasets and the two LLMs.

query that answers the query as the new query for retrieval. After obtaining the new queries through the aforementioned two methods, the BGE-Base and GTE-Base are utilized as retrievers to accomplish the RAG task.

5.1.4 Implementation Details

We employed the HuggingFace Transformers (Wolf et al., 2020) in PyTorch for the experiments. Except for the experiment in Sec. 5.4 on the impact of the number of retrievals on robustness, in all RAG tasks, three documents are retrieved for each query and we set the maximum generation length to 1024 for the generation models. Following (Jin et al., 2024), we test 1000 queries for each RAG dataset. For the training of CL models in Sec. 4.1, we set the learning rate to 2e-5, batch size to 32, and max sequence length to 256. For BERT in Sec. 4.2, we chose BERT-base-uncased for training, adopted the Adam (Kingma and Ba, 2014) optimizer, set the learning rate to 1e-8, and the batch size to 32. All experiments are conducted on Nvidia A6000 GPUs. More details can be found at the link provided in the Evaluation part of Sec. 3.

5.2 Main Results

Table 2 shows the main experimental results of different methods in six R^2AG datasets with three different corrupted query proportions (80%, 40%, 0%) when the retrieval model is BGE. From the table, we can draw the following observations:

Robustness against Corrupted Queries. In the case of the proportion of corrupted queries (80%, 40%), the existing retrieval models (BGE-Base) demonstrate an inability to proficiently perform RAG tasks. After standard contrastive learning training (BGE-CL), consistent improvements have been achieved in in-domain and cross-domain RAG tasks compared to the base retrieval model, and the improvement becomes more significant as the error ratio increases. On average, improvements of 2.2% and 7.1% are achieved at 40% and 80% error query ratios, respectively. This indicates that using the contrastive learning-based approach with the training dataset we proposed can effectively improve the robustness of the RAG model.

Analysis of Query Rewriting Methods. For query rewriting methods (Correct and HyDE), we can observe that the performance is not satisfactory in the cases of three different corrupted query proportions, and most of them are lower than the results obtained by using the base model for retrieval. This is because if the input query is corrupted, HyDE allows the LLM to generate a document answering based on the wrong query, which may be even more wrong because it may not know how to answer. LLM's correction performs relatively better than rewriting because the new query after correction will not deviate too far from the original query, unlike generating a document that might stray significantly. However, it is still lower than the method that uses the query directly for retrieval (BGE-Base), because LLM does not know how to judge the correctness of the query and may need additional information to know how to cor-

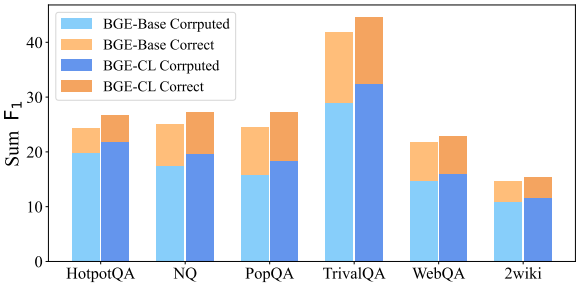


Figure 3: The contribution of correct and corrupted queries to the total F_1 score when the retrieval model is BGE-Base and BGE-CL, and the generative model is Llama3, with the F_1 contribution of correct queries stacked on top of the F_1 contribution of corrupted queries. The number of correct and corrupted queries for each model is the same.

rect the query. The correction still modifies the query in the wrong direction. Therefore, both of the above methods have difficulty in improving the robustness of the RAG model. In addition, performing query correction and query rewriting requires an additional step on top of the standard RAG to call LLM, which is costly. Therefore, we suggest improving the robustness of RAG by training an end-to-end retrieval model like Sec. 4.1, rather than through query rewriting.

Analysis of Standard Contrastive Learning. In the scenario with corrupted queries, standard contrastive learning can improve the robustness of the RAG model, but in the scenario where the error query ratio is 0 (original query), standard contrastive learning results in a decrease in the performance of the RAG model, both in in-domain and cross-domain. This is because when training with standard contrastive learning, using corrupted queries can retrieve the correct documents, but the problem that may come with this is that using correct queries may retrieve the wrong documents, leading to a decrease in the performance of the RAG model in the correct query scenario. We propose the adaptive retrieval model selection to solve this problem in Sec. 4.2 and conduct corresponding experiments in Sec. 5.5.

Dataset Specific Findings. We find that for some specific datasets, such as WebQA, the RAG method does not improve the performance, which is similar to the results in (Jin et al., 2024). There are mainly two issues. First, no relevant documents are retrieved. This is because the retrieval corpus (Wikipedia data from December 2018) may not cover the answer to the question or the retrieval model is not powerful enough. Second, LLM is sensitive to the prompt (Mizrahi et al., 2024) and irrelevant information retrieved brings noise to the LLM generation process. In addition, we find that due to the different difficulty levels of the questions, the correction results of LLM are also different. For example, NQ belongs to naturally simple questions, and LLM can correct them well, thereby improving the RAG performance. But overall, error correction still can't achieve satisfactory results.

We conduct experiments in Appendix A when the retrieval model is GTE and obtained conclusions similar to the above analysis.

5.3 Performance Comparison of Correct and Corrupted Query

The main experiments in Table 2 show the overall RAG performance of all queries (correct and corrupted queries), but we do not know how the RAG model performs for correct queries and corrupted queries separately. Therefore, in this section, we explore the contribution of correct and corrupted queries to the total F_1 of BGE-CL and BGE-Base respectively on six datasets with the corrupted query ratio of 80%. Here we have a total of 1000 queries, with 200 correct queries and 800 corrupted queries.

The results are shown in Figure 3. It can be seen that in all six datasets, the proportion of corrupted queries contributed by the BGE-CL is higher than that contributed by the BGE-Base method (the height of the bottom blue color), while the proportion of correct queries contributed by the two methods is not much different. This indicates that after training with standard contrastive learning on a retrieval dataset containing corrupted queries, the robustness of the RAG model to corrupted queries can be effectively improved in both in-domain and cross-domain RAG datasets, thereby improving the overall performance of the RAG model. This further verifies that when choosing to improve the robustness of the RAG model, we can use our provided training retrieval dataset to train a retrieval model end-to-end.

5.4 The Effect of the Number of Documents Retrieved

In standard RAG settings, retrieving different numbers of documents will have different impacts on the performance of RAG (Jin et al., 2024). Retrieving fewer documents makes it difficult for the generation model to find the answer to the query

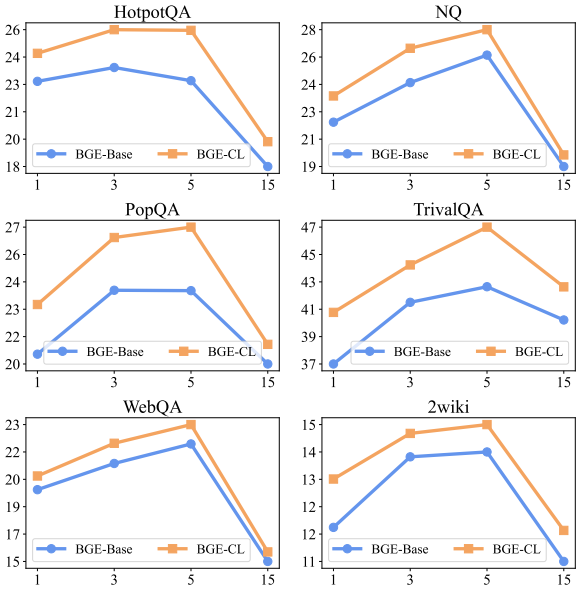


Figure 4: The results of BGE-Base and BGE-CL retrieving varying numbers of documents on six datasets when the LLM is Llama3. The x-axis represents the number of retrieved documents, specifically 1, 3, 5, and 15, while the y-axis indicates the token-level F_1 score.

Dataset	HotpotQA	NQ	TrivalQA	2wiki
	Llama3			
BGE-CL	36.46	40.24	61.60	20.07
ASR	37.65	42.45	62.67	20.54
	Qwen2			
BGE-CL	41.82	41.60	62.46	33.44
ASR	43.10	45.00	64.15	35.63

Table 3: The results of the ASR when the retrieval model is BGE in the scenario of 0% corrupted queries.

from the documents, while retrieving too many documents will expose the generation model to too much noise in the documents, making it difficult to grasp the key points. Therefore, in this section, we explore the impact of retrieving different numbers of documents on the robustness of the RAG model. We use BGE-Base and BGE-CL as retrieval models, Llama3 as the generation model, and conduct experiments on six datasets with 80% corrupted queries. The results are shown in Figure 4. Several conclusions can be drawn from the figure:

(a) Regardless of the number of documents retrieved, BGE-CL achieves better performance than BGE-Base, indicating that BGE-CL has higher robustness and is not limited by the number of documents retrieved, i.e., it is effective in all resources (retrieving different numbers of documents).

(b) The performance first increases and then decreases with the change in the number of documents retrieved, which is similar to the phenomenon in the correct query scenario (Jin et al., 2024), indicating that we need to choose an appropriate number when retrieving documents, considering more noise situations, to balance resources and RAG performance.

5.5 Effectiveness of ASR

Similar to the analysis in Sec. 5.3, we look at which part of the query contributes to the final F_1 score. When the corrupted query ratio is 0, it is all contributed by the correct query. Therefore, the total F_1 of BGE-CL has decreased compared to BGE, that is, for some correct queries, BGE-CL has retrieved inappropriate documents, leading to a decrease in RAG performance. This indicates that the correct query has retrieved incorrect documents (i.e., documents that are not conducive to the completion of the RAG task by LLM).

In Sec. 4.2, we propose to select the retrieval model according to the correctness of the query. The results are shown in Table 3. It can be seen that ASR can effectively improve the performance on both in-domain and cross-domain datasets. In addition, interrupting the cos similarity transmission (while enabling corrupted queries to retrieve correct documents, it should also ensure that correct queries do not retrieve incorrect documents) commonly used in standard contrastive learning is a potential solution, which is the direction we want to explore in the future.

6 Conclusion

In this work, we present the first comprehensive investigation into the robustness of Retrieval-Augmented Generation (RAG) systems against query entry errors. To advance research in this area, we introduce R^2AG, a benchmark dataset that focuses on three common types of errors: keyboard proximity, visual similarity, and spelling. Our extensive experiments on R^2AG demonstrate that state-of-the-art RAG retrievers, large language models, and query reformulation methods exhibit poor robustness when handling corrupted queries. To address this limitation, we propose two novel solutions: a contrastive learning-based approach, which enhances retriever robustness but slightly reduces performance on correct queries; and an adaptive retrieval selector, which dynamically selects different retrievers for correct and corrupted queries, resulting in overall improved robustness.

7 Limitations

We believe that R^2AG can promote the development of the LLM and RAG fields, yet it still has the following limitations: First, R^2AG currently includes only six datasets for QA and Multi-Hop QA. In the future, we plan to expand the benchmark to encompass a wider range of RAG-related datasets, such as fact-checking (Petroni et al., 2021; Thorne et al., 2018), multiple-choice (Hendrycks et al.; Lin et al., 2022) tasks, and others for the community to test. Second, in this paper, we primarily tested the performance of RAG models under standard RAG scenarios. As the RAG landscape continues to evolve, many new RAG models (Asai et al., 2024; Kim et al., 2024) are emerging. Therefore, in the future, we also aim to apply more RAG methods to R^2AG to test their robustness.

References

Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. 2023. Information retrieval meets large language models: a strategic report from chinese ir community. *AI Open*, 4:80–90.

AI@Meta. 2024. Llama 3 model card.

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

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Sunhao Dai, Yuqi Zhou, Liang Pang, Weihao Liu, Xiaolin Hu, Yong Liu, Xiao Zhang, Gang Wang, and Jun Xu. 2024. Neural retrievers are biased towards llm-generated content. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 526–537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023a. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. *arXiv preprint arXiv:2405.13576*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Jaehyung Kim, Jaehyun Nam, Sangwoo Mo, Jongjin Park, Sang-Woo Lee, Minjoon Seo, Jung-Woo Ha, and Jinwoo Shin. 2024. Sure: Summarizing retrievals using answer candidates for open-domain qa of llms. *arXiv preprint arXiv:2404.13081*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Yi Liu, Lianzhe Huang, Shicheng Li, Sishuo Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Recall: A benchmark for llms robustness against external counterfactual knowledge. *arXiv preprint arXiv:2311.08147*.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 7.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt llm evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. 2021. Kilt: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2024. Ares: An automated evaluation framework for retrieval-augmented generation systems. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 338–354.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819.

SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. 2024. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2023. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*.

Haike Xu, Zongyu Lin, Yizhou Sun, Kai-Wei Chang, and Piotr Indyk. 2024. Sparsecl: Sparse contrastive learning for contradiction retrieval. *arXiv preprint arXiv:2406.10746*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

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

Dezhi Ye, Bowen Tian, Jiabin Fan, Jie Liu, Tianhua Zhou, Xiang Chen, Mingming Li, and Jin Ma. 2023. Improving query correction using pre-train language model in search engines. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2999–3008.

A Main Results of GTE

The main results when the retrieval model is GTE are shown in Table 4. From the table, we can get conclusions similar to Section 5.2.

Dataset	HotpotQA	NQ	PopQA	TrivalQA	WebQA	2wiki	HotpotQA	NQ	PopQA	TrivalQA	WebQA	2wiki	Overall
Method	**Llama3**						**Qwen2**						
	80% Corrupted Queries												
Naive Generation	18.00	19.03	**13.32**	**41.51**	**26.64**	**24.89**	19.77	15.60	13.03	35.84	**24.60**	17.86	22.51
GTE-Base	17.66	19.39	5.61	35.01	14.70	10.58	24.94	20.84	9.19	39.00	19.81	21.66	19.86
GTE-CL	**21.96**	20.50	10.57	40.85	19.63	13.34	**27.42**	**21.09**	**13.06**	**43.13**	23.08	**25.46**	**23.34**
Correct	16.51	**22.10**	7.32	34.18	18.85	11.27	25.46	16.50	10.57	28.83	18.36	20.14	19.17
HyDE	6.12	14.54	1.80	20.10	14.70	3.71	7.96	14.15	5.37	22.59	13.46	6.72	10.94
	40% Corrupted Queries												
Naive Generation	23.11	24.37	**15.68**	**49.29**	**31.94**	**23.63**	24.21	21.67	15.30	42.28	28.20	27.63	27.27
GTE-Base	23.54	28.54	9.10	44.24	22.18	13.90	29.46	**30.90**	13.20	46.64	26.84	26.39	26.24
GTE-CL	**25.38**	**29.53**	13.07	47.49	26.71	15.86	**32.65**	30.47	**16.13**	**50.89**	**28.82**	**29.34**	**28.86**
Correct	15.96	26.67	8.03	31.95	21.68	11.80	18.90	19.22	13.81	27.89	19.58	21.18	19.72
HyDE	5.03	17.46	1.79	19.77	13.35	4.93	10.12	15.16	5.73	25.26	14.31	7.14	11.67
	0% Corrupted Queries												
Naive Generation	26.32	30.55	**19.95**	56.68	**37.98**	**29.00**	26.40	24.99	18.39	47.69	33.84	34.09	32.16
GTE-Base	29.54	**36.70**	11.81	56.67	28.67	15.79	37.18	**39.04**	17.18	58.31	33.02	31.48	32.95
GTE-CL	**32.28**	34.70	15.75	**56.97**	31.39	18.52	**39.58**	35.88	**19.56**	**59.94**	**34.51**	**33.59**	**34.39**
Correct	17.63	29.30	8.98	30.72	23.47	11.80	20.16	19.40	16.47	27.17	20.13	20.61	20.49
HyDE	6.84	19.41	1.44	21.04	12.82	5.42	11.88	16.32	5.33	28.39	17.18	6.46	12.71

Table 4: The overall performance of the RAG task under six datasets and three different error proportions of queries scenarios when the retrieval model is GTE, and the generator models are Llama3 and Qwen2. The optimal results are presented in bold. The **"overall"** column represents the average result of that row, which is the average result of the method across all datasets and the two LLMs.