From RAGs to Riches: How Search and Q&A Capabilities with RAG can Accelerate Engineering and Marketing Workflows

Ethan Moody, April 2024

Summary Statement

Retrieval-Augmented Generation (RAG) systems have the potential to enhance both search and question-answering capabilities by supplying timely and accurate information to users. This report details the specifications, performance, limitations, and future opportunities for such a system as a new potential resource for the company. A proof of concept (POC) test revealed that a RAG model can offer answers to domain-specific questions that are generally relevant, factually correct, and tailored to specific team needs, all at an impressive speed. We recommend that company leadership take the following three actions to capitalize on the benefits of this system for engineers and support staff: (1) build on the success of the POC by allocating resources to RAG model refinement and infrastructure development through a series of sprints over the next two quarters, (2) establish a process to periodically update the model's document store to prevent obsolescence, and (3) target full-scale deployment of the model by year-end.

Introduction: A RAG proof of concept (POC)

The debut of Retrieval-Augmented Generation (RAG) systems in 2020 marked a significant step forward in Generative AI. RAG systems supplement LLMs with dynamic document retrieval, enabling them to return "more specific, diverse, and factual language" in language generation tasks than pre-trained LLMs alone.[1] In this report, we explore the potential benefits of RAG for the company based on a limited proof of concept (POC) model. The main goal of our POC is to enhance the company's capabilities in document search and Q&A. This is accomplished by (1) affixing a dynamic resource pipeline to an LLM, through which relevant contextual information is transmitted and incorporated into responses to user queries, and (2) capitalizing on common GenAI tools — like embedding models and LLMs — to provide rapid text generation services. We believe both (1) and (2) can support accelerated product development and marketing efforts across areas like research, testing, troubleshooting, communication, and innovation.

The POC consists of the following components: (A) a database of documents ("document store") that includes recent arXiv research papers on RAG and NLP, several question-answering/prompt engineering blog posts by applied AI researcher Lilian Weng, and a number of Wikipedia articles on GenAI, information retrieval, and LLMs; (B) a pre-trained LLM used for generating text responses; and (C) an embedding model that helps the model retrieve the most relevant contextual information from the document store to assist with answering user questions. Together, these components ensure the RAG system can handle a wide array of user questions related to NLP and GenAI, including both highly technical and highly conceptual requests around topics like language model architectures, AI ethics, or normalized subspace similarity comparisons. Our POC approach also caters to the unique needs of our engineering and marketing staff. The model requires only a question as an input — like *"What is the main goal of prompt engineering in language models?"* — and then uses separate prompts to tailor its answers to these two groups. The "engineering" prompt asks the model to deliver detailed, technical answers that are rich in data, context, and examples. Meanwhile, the "marketing" prompt asks the model to generate more concise responses that provide a higher-level understanding of key concepts. This customization ensures that both supported user groups receive information that is well-calibrated to their respective job functions and levels of expertise.

Key Take-Aways: Five insights from the POC

- **Choice of language model (LLM) and embedding size matters:** In our POC, we explored two LLMs — one open-source model with ~7B parameters (`mistralai/Mistral-7B-Instruct-v0.1`, known as Mistral) and one proprietary model with even more parameters (known as `ChatCohere`) — and three embedding models with dimensions of 384 and 768. The superior performance of `ChatCohere` and the `avsolatorio/GIST-Embedding-v0` embedding model (with 768 dimensions) highlighted the connection between LLM parameter size and RAG response quality (more parameters is usually better), as well as the connection between embedding dimension size and effective query/document matching (higher dimension size is usually better).

[1]Lewis, P., et al. (2021, April 12). Retrieval-augmented generation for knowledge-intensive NLP tasks. arXiv.org. https://arxiv.org/abs/2005.11401

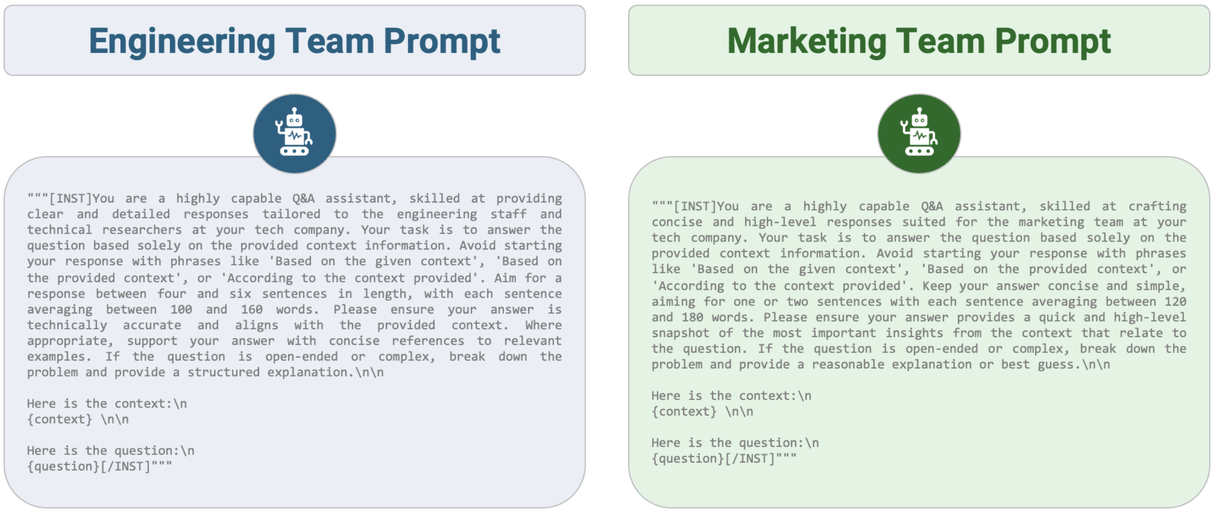
- **Chunking strategy impacts output quality:** We also tested a variety of document chunking (splitting) strategies, ranging from small, to medium, to large chunk sizes, each with some amount of overlap. A chunk size of 400 (medium) with an overlap of 20 ultimately produced better results than both smaller (e.g., 50) and larger (e.g., 500) sizes. Our main take-away here is that chunking can have a meaningful impact on the relevance and focus of retrieved content for RAG answers and can directly influence response output quality.
- **Simple retrieval methods work well "out of the box":** Several methods exist for RAG document storage, retrieval, and similarity scoring/ranking. For this POC, we implemented a vector store-backed retriever and applied the "default" setting of similarity search for querying texts in the vector store. This proved to be an effective strategy, as testing showed that our system effectively matched and retrieved relevant documents for most user queries. Evaluating the impact of more advanced search and retrieval methods — such as Maximum Marginal Relevance (MMR) — or employing similarity score thresholds to limit document retrieval beyond the top-K (we used $K = 4$) most similar items from the vector store are two areas of additional research and testing that could potentially build on this POC.
- **Tailored prompt design increases answer faithfulness:** While a one-size-fits-all prompt works for certain use-cases, we found that the implementation of separate prompts for engineering and marketing users was key to the success of this POC. A dual-prompting strategy yielded RAG-generated responses that were generally well-suited to each group's unique needs (e.g., technical and detailed where appropriate vs. not) and aligned with "ground truth" answers. This demonstrated the importance of persona-based customization in our system design and showcased the ability of LLMs to effectively tailor their output based on different instructions.
- **Retrieval systems are efficient, helpful, and fast:** Once a RAG pipeline is developed, document retrieval and answer generation require minimal time. In our testing, we found that the POC system's answers to example questions were typically relevant, accurate, and quick to generate (taking only a few seconds or less). Response times remained fast even when batch-testing numerous questions at a time (taking a few minutes). This underscores a key capability of RAG models to deliver pertinent information in a very timely fashion.

Methodology: Technical approach and evaluation strategy

We iteratively tested over ten different RAG model configurations to identify the best-performing system. We used a series of metrics to compare RAG performance across different LLMs, embedding models, document chunking parameters, and prompt strategies and arrive at our final technical specifications for the POC.

For the LLM, the POC features `ChatCohere`. While the exact number of trainable parameters for Cohere's model does not appear to be publicly disclosed, some estimates put it at 35-50B or more, which is substantially higher than the alternative `Mistral` LLM we tested. Generally, a higher number of trainable parameters enhances an LLM's ability to generate more nuanced and accurate responses, and we found that RAG performance improved when we used `Cohere` versus `Mistral`. We complemented this choice of LLM with the selection of `avsolatorio/GIST-Embedding-v0` for our embedding model. Of our three tested embedding models, the `GIST` model had the largest number of dimensions (768), allowing it to more effectively capture semantic nuances than smaller 364-dimension alternatives like `all-MiniLM-L6-v2`. To refine our retrieval process, we experimented with various document chunking strategies to balance context preservation/retention and retrieval efficiency. We ultimately selected a medium chunk size of 400 with an overlap of 20 between chunks, as this configuration proved effective in supporting context precision, context relevance, and overall efficiency. Finally, we experimented with different prompting strategies. A single prompt proved ineffective at addressing the unique information needs of our two core user groups (engineers and marketing/support staff), generally providing more-detailed, technical responses to queries rather than adjusting responses based on group. We ultimately developed two different prompts (shown in **`Figure 1`** below) to support these groups, essentially prioritizing depth over breadth for engineers and prioritizing breadth over depth for marketing staff. The variety of resources included in our document store allowed our POC system to generate highly technical and detailed answers when prompted to respond like a Q&A assistant for engineering staff and technical researchers, and to generate easily digestible summary-style answers when prompted to respond like a Q&A assistant for marketing staff.

We experimented with the above parameters by testing the RAG system against a list of NLP- and GenAI-related questions with pre-approved ("gold") answers and measuring its performance across several key metrics, many of which range from 0 (worst) to 1 (best). For each model iteration prior to our final run, we tested on a randomly selected subset of 15 questions from the full list (representing 20% of the list), using the same subset each time. In our final run, we tested the model on the entire 75-question set. Traditional NLP metrics like `BLEU` and `ROUGE`, along with more advanced frameworks including `BERTScore` and `RAGAS`, offered insights into the system's ability to generate contextually accurate, relevant, and reliable responses. We also analyzed RAG response lengths versus "gold" answer lengths and


Engineering Team Prompt

```
"""[INST]You are a highly capable Q&A assistant, skilled at providing
clear and detailed responses tailored to the engineering staff and
technical researchers at your tech company. Your task is to answer the
question based solely on the provided context information. Avoid starting
your response with phrases like 'Based on the given context', 'Based on
the provided context', or 'According to the context provided'. Aim for a
response between four and six sentences in length, with each sentence
averaging between 100 and 160 words. Please ensure your answer is
technically accurate and aligns with the provided context. Where
appropriate, support your answer with concise references to relevant
examples. If the question is open-ended or complex, break down the
problem and provide a structured explanation.\n\n

Here is the context:\n
{context} \n\n

Here is the question:\n
{question}[/INST]"""
```

Marketing Team Prompt

```
"""[INST]You are a highly capable Q&A assistant, skilled at crafting
concise and high-level responses suited for the marketing team at your
tech company. Your task is to answer the question based solely on the
provided context information. Avoid starting your response with phrases
like 'Based on the given context', 'Based on the provided context', or
'According to the context provided'. Keep your answer concise and simple,
aiming for one or two sentences with each sentence averaging between 120
and 180 words. Please ensure your answer provides a quick and high-level
snapshot of the most important insights from the context that relate to
the question. If the question is open-ended or complex, break down the
problem and provide a reasonable explanation or best guess.\n\n

Here is the context:\n
{context} \n\n

Here is the question:\n
{question}[/INST]"""
```

Figure 1: Prompts were tailored to the unique needs of user groups to improve RAG answer quality.

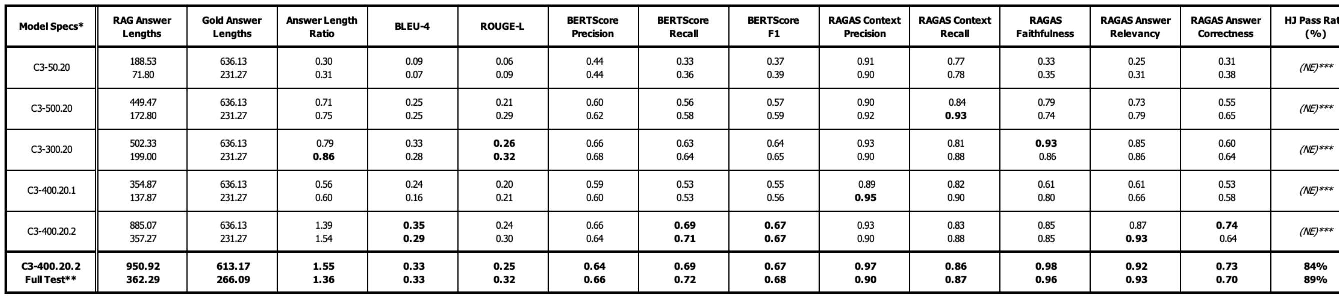
performed a manual review of RAG responses against the "gold" answers with our final model to incorporate an overall pass rate based on human judgment.

We used `BLEU-4`, `ROUGE-L`, and `Answer Length Ratio` (a comparison of average character counts for RAG-generated answers to reference answers for the same batch of questions) to measure the *syntactic* similarity between our RAG responses and "gold" answers. Each of these metrics examines the extent of token/word overlap — and, by extension, length differences — between RAG/"gold" answers. We quickly saw that RAG-generated answers could still be of high-quality while having low `BLEU` or `ROUGE` scores (e.g., <0.4) or a sub-optimal `Answer Length Ratio` (e.g., <0.8 or >1.2). While many of our RAG-generated answers had different phrasing than the "gold" answers, they generally still conveyed the same meaning. To address this limitation, we also computed `BERTScore` and `RAGAS` metrics, which provided a more helpful view of performance by assessing *semantic* similarity and context relevancy. We paid particular attention to the `BERTScore-F1` metric as a means of capturing both the completeness and relevance of RAG responses compared to "gold" answers based on contextual embeddings. Additionally, we examined `RAGAS-Faithfulness`, `RAGAS-AnswerRelevancy`, and `RAGAS-AnswerCorrectness` metrics to determine whether the generation component of our RAG system produced answers that were factually accurate, appropriate for the questions asked, well-supported by context information, and closely aligned with the "gold" responses. To ensure the retrieval component of our RAG system also functioned well, we monitored `RAGAS-ContextPrecision` — a measure of how effective the system is at selecting only relevant information to answer queries — and `RAGAS-ContextRecall` — a measure of how effective the system is at capturing all necessary information from the document store to construct correct and complete answers.

This framework provided a thorough assessment of the system's performance, capturing aspects like the literal accuracy of the generated answers, the alignment of these answers to ground truths, and the extent of context incorporation in the answers. Our selected POC model performed reasonably well across the most critical metrics, scoring between 0.67-0.68 for `BERTScore-F1` for the two different prompts, between 0.96-0.98 for `RAGAS-Faithfulness`, between 0.92-0.93 for `RAGAS-AnswerRelevancy`, and between 0.70-0.73 for `RAGAS-AnswerCorrectness`. The model also achieved scores >0.85 for `RAGAS-ContextPrecision` and `RAGAS-ContextRecall`, demonstrating acceptable performance on document retrieval. Although average RAG answer lengths were generally close to 1.5x as long as "gold" answer lengths, we were not concerned by this finding given the relatively short text being generated (~362 characters in length for marketing team answers and ~950 characters in length for engineering team answers) and the fact that users could always opt to further trim these outputs if desired. Additionally, our final manual review of RAG answers yielded a pass rate between 84-89%, indicating that more than four out of every five answers contained a sufficient level of detail and similarity to the "gold" answers. A summary of results for several key model configurations is shown in **`Figure 2`** below, and a more comprehensive definition of each metric is provided in the code notebook accompanying this report.

Results and Findings: RAG benefits, learnings, and limitations

Based on its performance, the selected RAG model is largely successful at fulfilling our POC goal. While search engines like Google already offer rapid question-answering services, the average person typically spends considerable time sifting through results to find the most pertinent information. A RAG system can address this issue by quickly assessing,



Model Specs*	RAG Answer Lengths	Gold Answer Lengths	Answer Length Ratio	BLEU-4	ROUGE-L	BERTScore Precision	BERTScore Recall	BERTScore F1	RAGAS Context Precision	RAGAS Context Recall	RAGAS Faithfulness	RAGAS Answer Relevancy	RAGAS Answer Correctness	HJ Pass Rate (%)
C3-50.20	188.53 71.80	636.13 231.27	0.30 0.31	0.09 0.07	0.06 0.09	0.44 0.44	0.33 0.36	0.37 0.39	0.91 0.90	0.77 0.78	0.33 0.35	0.25 0.31	0.31 0.38	*(NE)****
C3-500.20	449.47 172.80	636.13 231.27	0.71 0.75	0.25 0.25	0.21 0.29	0.60 0.62	0.56 0.58	0.57 0.59	0.90 0.92	0.84 **0.93**	0.79 0.74	0.73 0.79	0.55 0.65	*(NE)****
C3-300.20	502.33 199.00	636.13 231.27	0.79 **0.86**	0.33 0.28	**0.26** **0.32**	0.66 0.68	0.63 0.64	0.64 0.65	0.93 0.90	0.81 0.88	**0.93** 0.86	0.85 0.86	0.60 0.64	*(NE)****
C3-400.20.1	354.87 137.87	636.13 231.27	0.56 0.60	0.24 0.16	0.20 0.21	0.59 0.60	0.53 0.53	0.55 0.56	0.89 **0.95**	0.82 0.90	0.61 0.80	0.61 0.66	0.53 0.58	*(NE)****
C3-400.20.2	885.07 357.27	636.13 231.27	1.39 1.54	**0.35** **0.29**	0.24 0.30	0.66 0.64	**0.69** **0.71**	**0.67** **0.67**	0.93 0.90	0.83 0.88	0.85 0.85	0.87 **0.93**	**0.74** 0.64	*(NE)****
C3-400.20.2 Full Test**	**950.92 362.29**	**613.17 266.09**	**1.55 1.36**	**0.33 0.33**	**0.25 0.32**	**0.64 0.66**	**0.69 0.72**	**0.67 0.68**	**0.97 0.90**	**0.86 0.87**	**0.98 0.96**	**0.92 0.93**	**0.73 0.70**	**84% 89%**

Note: This table provides a non-exhaustive list of model metrics (over 10 different specifications were tested); the top numbers represent engineering answer scores, while the bottom numbers represent marketing answer scores; BLEU-1/2/3 and ROUGE-1/2/Lsum were also evaluated

*Model specs are abbreviated as LLM (C = Cohere, M = Mistral), Embedding Model (1 = all-MiniLM-L6-v2, 2 = multi-qa-mpnet-base-dot-v1, 3 = GIST-Embedding-v0), Chunk Size (e.g., 400), and Chunk Overlap (e.g., 20); the final C3-400.20 model (.2) featured improved prompts

**Full Test shows model results across entire set of validation questions and "gold" answers; all other rows show results for models tested on a 15-item (~20%) batch of validation questions and "gold" answers; model C3-400.20.2 represents the selected model for the RAG system

***(NE) stands for "Not Evaluated"; the Human Judgment (HJ) Pass Rate test was only calculated for the 75-question Full Test of the selected model for this POC given the time required to manually review RAG system-generated answers and contexts and perform this calculation

Figure 2: RAG model results highlight system's ability to provide acceptable answers to domain-specific questions.

ranking, and retrieving context-specific information from a relevant document store and integrating this content into generated answers. This is one of the key benefits of our POC: the model has quantifiably demonstrated an ability to return relevant and accurate answers to a broad range of domain-specific questions, which should speed up the process of information delivery to user groups. For engineers, this means more rapid researching, fact-checking, troubleshooting, and innovation during product development and quarterly releases. For marketing staff, this makes it easier and faster to distill down complex NLP/GenAI concepts to layman's terms, onboard and train new team members, and craft communications that engage target consumer markets. For both groups, RAG can ultimately support productivity, accelerating the pace at which typical workflows are completed.

Development and testing during the POC yielded a few valuable learnings. From a technical standpoint, the POC underscored the importance of "architectural tuning" — e.g., LLM and embedding model selection, chunk size determination, prompt design/persona customization, and efficient RAG pipeline development (using packages like `LangChain`) — for driving optimal RAG system performance. From a non-technical standpoint, the POC highlighted the potential for RAG models to serve as very capable, assistive text-generators — at least for specific tasks and for specific user groups with well-defined information needs. Most importantly, we recognize the system has a few key limitations. These include (1) the potential for generating inappropriate or harmful content given that neither the system as a whole nor the underlying LLM has been trained for safety, (2) the risk of factual inaccuracies or irrelevance in responses due to outdated information in the document store (or simply imperfect performance in general), (3) limited scalability to domains outside of NLP/GenAI, and (4) improper use as a tool for complex abstract reasoning or creative problem-solving tasks. Addressing these limitations and risks will be crucial prior to full-scale RAG deployment and equally important to keep front-of-mind for continuous model improvement post-launch.

Bottom-Line: Final recommendation and next steps

Despite these imperfections, we recommend a "live" (full-scale) implementation of a *refined version of the POC* at the end of the year. This would allow for six more months of model refinement through experimentation and testing, and would provide sufficient time to address many of the limitations of the current system. It would also enable product and IT leadership to perform a more complete assessment of key architectural and budgetary considerations prior to deployment, ranging from where to house the model, to average and peak load estimates (based on expected user engagement or pre-launch trials), to developer resource needs for maintaining the model in our production system, to cost plans (e.g., pay-per-use deployment, LLM reserve deployment, etc.) and miscellaneous expenses (e.g., API calls).

For next steps, we propose the following to further improve on the POC: (1) integration of a pipeline for automated updates to the document store (pulling in additional up-to-date content on GenAI research, methods, and state-of-the-art models) to prevent "context drift" and model irrelevance; (2) development of additional question-answer pairs (i.e., an extension of the question set used in model testing) for fine-tuning the LLM to domain-specific queries to further improve the accuracy and relevance of responses; (3) further exploration of chunking and retrieval strategies to boost model performance metrics; (4) further optimization of engineering and marketing prompts to limit inaccurate or sub-optimal responses (e.g., lengthier/more detailed than intended); (5) creation of a process for conducting ongoing scoring and analysis of RAG responses to assess model performance over time; and (6) translation of POC model code to a web app that allows for simpler user interaction with a cleaner UI (e.g., user types a question in a chat window and receives a response on the same page). With these improvements, our foundational RAG system will be able to offer even more benefits to the company and become a powerful new resource in the hands of engineering and marketing staff, helping them provide industry-leading products and services to our customers.

Appendix

Miscellany: Results of individual RAG model configurations and non-selected experimental runs/iterations, "pass"/"fail" judgments and corresponding reasoning for individual answers, as well as full lists of questions, answers, contexts, and source documents (for both model validation and test sets) are stored separately and can be supplied to the interested reader at request.

Additional Deployment Considerations & Average and Peak Load Estimates: Based on current company size, we estimate a fully-deployed RAG model would need to handle an average of 4-8 queries per staff member per day — with about 5-7 expected for each engineer (x300) and 2-3 expected for each marketing team member (x40). This translates to a total of ~8,000-11,000 queries per 5-day workweek. Peak load periods would likely occur during quarterly product releases, where engineers might be using the system more regularly for troubleshooting issues, supporting QC checks, or looking up technical details, and where marketing staff might be using it more regularly for drafting communications to support product launches. During these periods, we estimate system usage would roughly double to 8-16 queries per staff member per day (or to ~16,000-22,000 per workweek), with perhaps even additional volume (e.g., another 100-200 queries) expected over the weekend. Given this reasonably predictable variation in system load and the relatively low volume of potential users (340 current staff members in total), we recommend a pay-per-use deployment of the RAG model. This would allow the company to scale up resources during peak load periods, scale them down during average load periods, and optimize costs by paying for only what is needed by the staff without incurring any unnecessary fixed expenses.

Supplementary Charts: The below charts provide visuals for certain concepts/metrics discussed in the report.

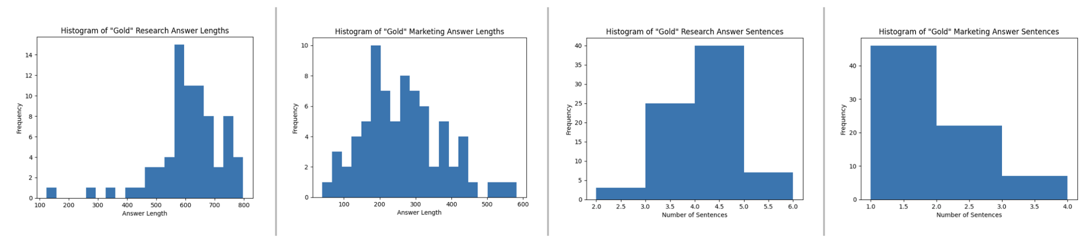


Figure 3: Answer length and sentence length distributions for 'gold' answers.

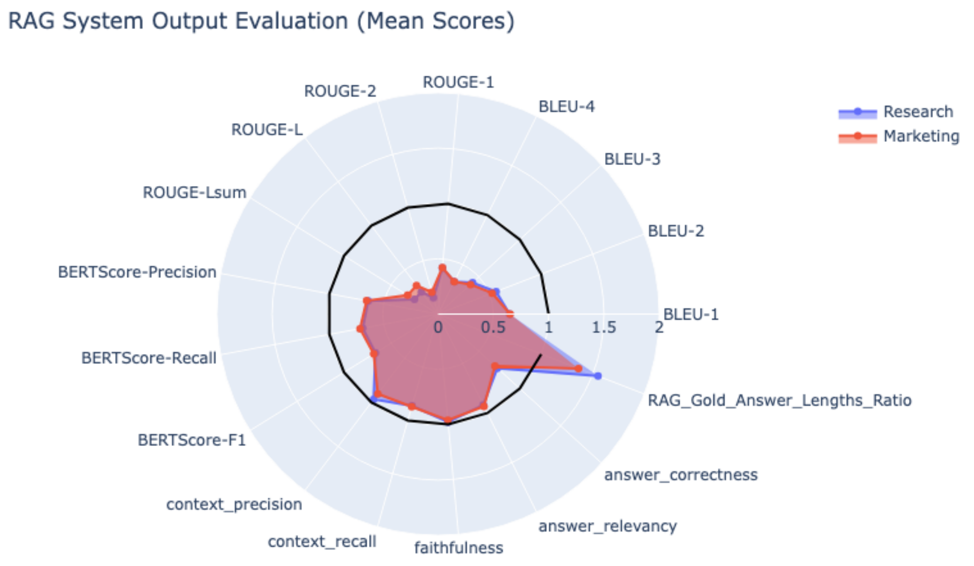


Figure 4: RAG POC results by evaluation metric, visualized (scatterpolar plot).