HiPerRAG: High-Performance Retrieval Augmented Generation for Scientific Insights

Ozan Gokdemir[1,2†*], Carlo Siebenschuh[1,2†], Alexander Brace[1,2†], Azton Wells[1†], Brian Hsu[1,2†], Kyle Hippe[1,2], Priyanka V. Setty[1,2], Aswathy Ajith[2], J. Gregory Pauloski[2], Varuni Sastry[1], Sam Foreman[1], Huihuo Zheng[1], Heng Ma[1], Bharat Kale[1], Nicholas Chia[1], Thomas Gibbs[3], Michael E. Papka[1,4], Thomas Brettin[1], Francis J. Alexander[1], Anima Anandkumar[5], Ian Foster[1,2], Rick Stevens[1,2], Venkatram Vishwanath[1], Arvind Ramanathan[1,2],

[1]Argonne National Laboratory, Lemont, Illinois, USA [2]The University of Chicago, Chicago, Illinois, USA
[3]NVIDIA Inc., Santa Clara, California, USA [4]University of Illinois Chicago, Chicago, Illinois, USA
[5]California Institute of Technology, Pasadena, California, USA
†Joint first authors *Corresponding authors: ramanathana@anl.gov, stevens@anl.gov, venkat@anl.gov

Abstract

The volume of scientific literature is growing exponentially, leading to underutilized discoveries, duplicated efforts, and limited cross-disciplinary collaboration. Retrieval-Augmented Generation (RAG) offers a way to assist scientists by improving the factuality of Large Language Models (LLMs) in processing this influx of information. However, scaling RAG to handle millions of articles introduces significant challenges, including the high computational costs associated with parsing documents and embedding scientific knowledge, as well as the algorithmic complexity of aligning these representations with the nuanced semantics of scientific content. To address these issues, we introduce HiPerRAG, a RAG workflow powered by high performance computing (HPC) to index and retrieve knowledge from more than 3.6 million scientific articles. At its core are Oreo, a high-throughput model for multimodal document parsing, and ColTrast, a query-aware encoder fine-tuning algorithm that enhances retrieval accuracy by using contrastive learning and late-interaction techniques. HiPerRAG delivers robust performance on existing scientific question answering (Q/A) benchmarks and two new benchmarks introduced in this work, achieving 90% accuracy on SciQ and 76% on PubMedQA—outperforming both domain-specific models like PubMedGPT and commercial LLMs such as GPT-4. Scaling to thousands of GPUs on the Polaris, Sunspot, and Frontier supercomputers, HiPerRAG delivers million document-scale RAG workflows for unifying scientific knowledge and fostering interdisciplinary innovation.

CCS Concepts

• **Information systems → Question answering**; **Similarity measures**; **Language models**; **Information extraction**; • **Computing methodologies → Learning to rank**.

Keywords

HPC, AI, Large Language Models, Retrieval-Augmented Generation, Metric Learning, Neural Information Retrieval

ACM Reference Format:
Ozan Gokdemir[1,2†*], Carlo Siebenschuh[1,2†], Alexander Brace[1,2†], Azton Wells[1†], Brian Hsu[1,2†], Kyle Hippe[1,2], Priyanka V. Setty[1,2], Aswathy Ajith[2], J. Gregory Pauloski[2], Varuni Sastry[1], Sam Foreman[1], Huihuo Zheng[1], Heng Ma[1], Bharat Kale[1], Nicholas Chia[1], Thomas Gibbs[3], Michael E. Papka[1,4], Thomas Brettin[1], Francis J. Alexander[1], Anima Anandkumar[5], Ian Foster[1,2], Rick Stevens[1,2], Venkatram Vishwanath[1], Arvind Ramanathan[1,2], . 2025. HiPerRAG: High-Performance Retrieval Augmented Generation for Scientific Insights. In *Platform for Advanced Scientific Computing Conference (PASC '25), June 16–18, 2025, Brugg-Windisch, Switzerland.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3732775.3733586

1 Introduction

Over the past century, the volume of scientific publications has increased rapidly. Today, this trend continues at an unprecedented pace. For example, the National Science Foundation reported a 50-fold increase in open-access scientific publications between 2013 and 2022 [22]. In the biomedical field alone, PubMed processed approximately 1.69 million articles last year, averaging more than three articles per minute [45]. Despite this vast output, researchers' capacity to keep up is limited, estimated at 22 articles per month [74]. This overload of information has significant consequences: more than half of the 200 million scholarly papers available today have never been cited [67]. Meanwhile, the top 1% of most cited scientists cumulatively receive 21% of all citations [57]. This imbalance leads to a lack of diversity in academic ideas, redundant research efforts, and hindered scientific progress [20]. Addressing these challenges requires innovative approaches to ingesting and disseminating scientific knowledge more effectively. In particular, machine learning poses a promising path towards helping researchers navigate this rapid influx of information.

The advent of large language models (LLMs) has provided researchers with performant tools for processing vast scientific corpora. LLMs have been explored to help with knowledge-intensive tasks, such as question answering [7, 10], information retrieval [7], document summarization [17, 31], and generating novel hypotheses [13]. Despite their utility, a crucial complication emerges when LLMs are applied to science, mainly, the need for any generated output to be grounded in factuality. Unfortunately, LLMs are prone to hallucinate plausible yet factually incorrect information, and science-based LLMs showcase limitations that prevent their wide-scale adoption in the scientific enterprise [32, 70]. In particular, enforcing factuality and reducing hallucination is an ongoing challenge involving multiple avenues of research [9, 21, 30].

The introduction of Retrieval-Augmented Generation (RAG) [48] aims to mitigate hallucinations of LLMs through the integration of neural information retrieval with LLM-based content generation. This approach leverages the ability of LLMs to represent any chunk of text—a word, a sentence, a paragraph, or a whole document—as a fixed-size embedding vector that encodes semantic relationships akin to meaning. The distance between these embedding vectors is used as a measure of relevance between two texts such as a question and a scientific passage answering it. Quantifying relevance in this manner improves the factuality of LLM outputs by dynamically equipping LLMs with contextually relevant passages during the generation process [39].

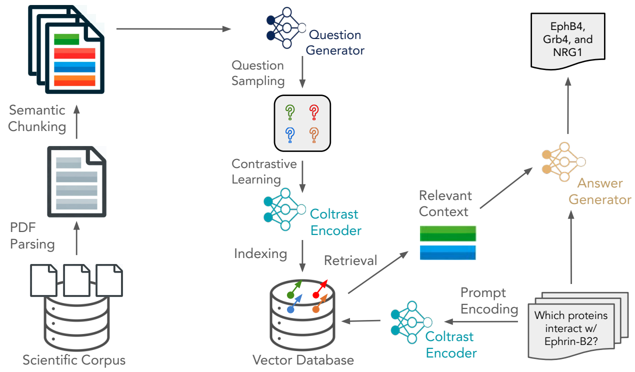

Figure 1: HiPerRAG Workflow. A graphical overview of our implementation of RAG for scientific literature. We implement a novel PDF parsing approach, namely optical recognition with eclectic output (Oreo) that takes into account intrinsic formatting of multi-layout scientific manuscripts. We then design a query-aware encoder finetuned on scientific literature which enables to semantically organize the data into relevant chunks that can be used to retrieve the most relevant data on a per-query basis. Both these innovations enable us to achieve both computational efficiency and retrieval performance at scale.

Despite the widespread adoption of RAG, it faces three significant technical challenges that hinder its ability to scale to millions of documents. These include:

(1) **Parsing meaningful and coherent text from scientific documents:** This challenge emerges from the diverse organization semantics of how papers are published, including dense arrangements of tables, figures, and equations in a portable document format (PDF) file. This process requires intricate subtasks that range from image normalization to character recognition, making it a complex endeavor.
(2) **Optimizing the retrieval accuracy using LLMs:** LLMs need to be provided with the most relevant information—and only that—to answer the specific query at hand. This crucial task depends on an encoder's ability to produce vector representations of both user queries and the document corpus. General purpose encoders perform suboptimally when encoding scientific content since such content constitutes only a small portion of mainstream LLM training datasets [28, 68].
(3) **Evaluating RAG systems on scientific literature:** Scientific literature carries unique features including domain-specific vocabulary, empirical evidence, and citations, which are generally hard to assess. Hence this imposes certain restrictions on how RAG can be evaluated on scientific literature, where fetching both the right context and the appropriate related content can be difficult.

Contributions: Given these limitations, our work, HiPerRAG[1] embodies innovations on several components of the RAG framework for scientific literature, including document parsing, encoding scientific information, and science-specific evaluation benchmarks to overcome each of the aforementioned challenges (see Figure 1). Our contributions are summarized as follows:

- We develop neural Optical Recognition with Eclectic Output (Oreo), a layout-aware multimodal scientific document parser for high-quality content extraction from PDFs. Oreo outperforms state-of-the-art parsers in terms of throughput while maintaining a commensurate accuracy. Section 3.1 and Section 4.1 discuss Oreo's design, implementation, and evaluation.
- We design a query-aware encoder tailored for high accuracy on scientific content that is both state-of-the-art and performs at scale. To accomplish this we develop a novel finetuning algorithm for scientific text encoding, called ColTrast, which combines contrastive and late-interaction methods to capture the benefits of both to custom-tailor passage retrieval to user queries. Section 3.2 presents ColTrast's design and implementation and Section 4.2 presents its evaluation results.
- We introduce two new biomedical Q/A benchmarks for RAG containing question-answering pairs for protein interaction and function predictions, namely the ProteinInteractionQA (7591 Q/A pairs) and ProteinFunctionQA (17,646 Q/A pairs). Section 4.3 discusses their curation in detail. We also introduce a synthetic dataset, BioSynthQPs, consisting of 1500 domain-specific scientific passages, to evaluate the retrieval accuracy of encoders. Section 4.3 describes the process of its creation.
- We leverage the Parsl [11] framework to distribute document parsing, encoding, and retrieval components of a RAG workflow to an arbitrary number of nodes on diverse HPC systems. Section 5 contains scaling experiments on the Polaris, Sunspot, and Frontier supercomputers.

The remainder of this paper is organized as follows. Section 2 surveys the current state of the art in document parsing, neural information retrieval, and retrieval-augmented generation. Section 3 presents the detailed design and implementation of the individual components of HiPerRAG. Section 4 evaluates the HiPerRAG system, detailing the performance of the OREO parser and ColTrast encoder individually, and the overall efficacy of the system on scientific Q/A tasks. It also introduces the Q/A and retrieval datasets used for testing. Section 5 discusses the performance attributes of HiPerRAG and presents the scaling experiments on three supercomputers. Section 6 concludes the paper with a discussion of current limitations, future research directions, and implications.

[1] https://github.com/ramanathanlab/distllm

2 Background

We survey related work on document parsing, neural information retrieval, metric learning, and retrieval-augmented generation.

2.1 Document Parsing

The portable document format (PDF) has become the primary mode of scientific communication [6]. Although PDFs can store diverse content including figures, tables, equations, and references, they are not machine-readable. [1]. Therefore, the content of a PDF must be converted into structured outputs, such as raw text, tabular data, and figures. We refer to this task as document parsing.

Extraction-based parsers such as PyPDF [5], PyMuPDF [4], and PDFMiner [3] leverage the internal structure of PDF files to obtain text and metadata. However, while these methods achieve high throughput, they rely on an embedded text layer that is generally only present in born-digital PDFs, i.e., they do not apply to scanned PDFs. Furthermore, their accuracy is sensitive to formatting errors within PDF files, resulting in a high file-level failure rate and low overall accuracy. Finally, extraction-based parsing is mostly restricted to raw text, neglecting tables and figures.

Optical Character Recognition (OCR) is a more advanced document parsing strategy that generates machine-readable content directly from the image renderings of the PDF pages. Traditional OCR approaches involve several deterministic steps such as image pre-processing (including noise reduction and normalization), segmentation (to separate characters or words), feature extraction (to identify distinctive character features), and eventually culminating in character recognition and extracted text. With the advent of deep learning in computer vision, the use of neural networks has streamlined this process either by replacing portions of the OCR pipeline [80] or by training an end-to-end image-to-text decoder [14, 43]. The leading edge in document parsing is Neural Optical Understanding for Academic Documents (Nougat) [14] that uses a Vision Transformer (ViT) to embed a document page image and a text decoder to generate text from it. Nougat employs a hierarchical ViT to process large-image renderings of PDF pages, essentially encapsulating both low-resolution OCR tasks such as layout identification and high-resolution ones such as character recognition in one end-to-end network architecture. Since hierarchical ViTs learn to decode the contents of a page directly from its image, they require GPU-accelerated training on large high-quality datasets. This requirement introduces scalability challenges. In addition, despite being ViT-based, these approaches are currently unable to process data modalities other than text, such as figures.

To achieve streamlined ViT-based document parsing, recent methods such as Marker [60] explored a hybrid of layout-based and OCR-based parsing. Marker first performs layout detection to segment the PDF into text, image, table. These segments are then parsed using the Texify [59] pre-trained ViT to extract machine-readable content. This model claims to realize an approximately 10-fold improvement over Nougat [14], which we could not confirm in our experiments.

We propose Optical Recognition with Eclectic Output (Oreo), which substantially accelerates document parsing while maintaining comparable accuracy to that of leading-edge parsers. Oreo achieves this by conducting PDF parsing in two stages. First, it leverages a convolutional neural network architecture (YOLO) for layout detection [63]. Second, it applies Texify to regions predicted to be text to retrieve textual content. Oreo achieves a 4.5× speedup and a 94.6× improvement in FLOP utilization over current state-of-the-art (SOTA) methods, including Marker. It supports multimodal parsing for text, figures, and tables, and maximizes GPU utilization to deliver the highest throughput with comparable accuracy to other vision-based parsers. This efficiency highlights Oreo's capability for targeted content extraction and superior hardware resource leverage.

2.2 Neural Information Retrieval and Metric Learning

Neural information retrieval uses compact vector representations of data, known as embeddings, that capture meaningful characteristics [39, 55]. The distance between the embeddings of a query and a document, measured by cosine similarity or Euclidean distance, quantifies their relevance to each other. This measurement allows retrieval systems to efficiently identify and retrieve documents that are most relevant to a specific query by comparing these distance metrics. Since retrieval results depend directly on the embedding vectors, training on a broad scientific text corpus is a prerequisite for an encoder LLM to offer embeddings suitable for accurate retrieval on scientific content.

While Transformer-based LLMs are successful information encoders [51], most general-purpose LLMs are not exposed to enough domain-specific scientific data during training [28, 68]. Since academic writing style differs substantially from general data such as news articles, embeddings produced by these LLMs are suboptimal for accurate information retrieval for niche scientific queries. Previous work has enhanced the performance of LLMs on downstream tasks by pre-training encoders from scratch using domain-specific data. This approach has been applied across various fields, including biomedical [33, 47], chemical [65, 75, 84], and material science [72]. While domain-specific training of encoders has proven valuable, incorporating metric learning techniques can further refine retrieval performance by optimizing the encoder to better align query embeddings with the most relevant document embeddings.

Several techniques have been explored for building boutique encoders for domain-specific scientific content. In particular, contrastive learning [46] has improved retrieval accuracy by better estimating the relevance of document passages to queries. Contrastive learning teaches models to distinguish between similar (positive) and dissimilar (negative) pairs, encoding data such that positives are closer and negatives farther apart in feature space. Methods like COSTA [52] and FILIP [81] successfully applied this to text and image encoders, respectively.

Late interaction, introduced in ColBERT [41], a BERT [40] variant, computes token-level contextualized embeddings by maximizing pairwise similarity between query and text token embeddings. This fine-grained comparison increases semantic alignment and retrieval accuracy, but is computationally expensive, replacing a single tensor operation with N^2 tensor operations. ColBERTv2 [64] optimizes this by performing token-wise comparisons only after encoding, enabling offline computation of document embeddings and efficient retrieval using a vector database. While this improves efficiency, it relies on ad-hoc re-ranking and does not scale to end-to-end ranking for large corpora. In this work, we propose the ColTrast algorithm, which combines the late interaction mechanism and contrastive learning to achieve robust retrieval accuracy and scales to thousands of GPUs to cater for millions of documents.

2.3 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a prominent technique that integrates neural information retrieval methods with large language models (LLMs) to enhance the factuality and relevance of generated outputs. In their seminal work, Lewis et al. [48] demonstrated the effectiveness of RAG in tasks such as question answering, fact-checking, and knowledge extraction. Introduced at a time when LLMs were already extensively adopted in industry and academia, and their limitations in factuality, relevance, and bias were well-documented, RAG represented a significant advancement in Natural Language Processing.

A key preliminary step in implementing a RAG application involves chunking and encoding a curated corpus of relevant documents. The resulting chunk embeddings are stored in a vector database such as Faiss [24] or Chroma [69], each linked to a unique ID that maps to the raw text in a separate database. When a query is provided to an LLM, it is encoded using the same encoder as the document chunks, and the resulting query embedding is used for a semantic search in the vector database to identify the nearest neighbors. Common distance metrics for this search include cosine similarity and Euclidean distance. Finally, the retrieved IDs are used to fetch the raw content of the relevant documents, which are supplied to the LLM alongside the query to generate a response.

RAG has been widely adapted to many scientific domains, including material science [56], biology [53, 76], and chemistry [18]. However, scaling a RAG application up to millions of documents remains an ongoing challenge in both academia and industry. In response, we present HiPerRAG, a high-performance computing infrastructure designed to scale every component of RAG to an arbitrary number of compute nodes. With 3.6 million indexed scientific papers, HiPerRAG represents a step toward integrating approximately 200 million existing scientific papers into a single platform for scientific Q/A, literature review and hypothesis generation.

3 HiPerRAG Design and Implementation

Despite the continued advancement of Large Language Models (LLMs) in reasoning and creative writing, their limitations in operations like tool usage or information retrieval persist. This has spurred the development of sophisticated software stacks that enhance LLMs with external tools, such as information retrievers, to augment text generation processes and implement Retrieval-Augmented Generation (RAG) applications [26, 48, 79]. In response to the scalability challenges in existing RAG frameworks, we introduce HiPerRAG—an HPC software stack designed to build scalable, million-document RAG applications for scientific use. HiPerRAG serves not as a standalone model, but as an infrastructure that supports the development of one of the most comprehensive RAG systems on scientific data to date.

To enhance the scalability and accuracy of RAG, our work introduces innovations in PDF parsing and encoder fine-tuning through Oreo and ColTrast, respectively. In Section 3.1 we discuss the implementation details of Oreo. Section 3.2 presents the ColTrast algorithm. Section 3.3 will conclude this section with the details of the overall HPC workflow that powers HiPerRAG and the warm-start optimization we implemented to optimize throughput and GPU utilization. Figure 1 depicts the high-level RAG workflow implemented by HiPerRAG.

3.1 Document Parsing with Oreo

A system that extracts text from PDFs follows one of two paradigms: the end-to-end approach or the compartmentalized approach. End-to-end document parsing generally relies on the use of multi-modal AI algorithms which, despite making use of GPU acceleration, are challenging to scale to large (>1M) document corpora due to the quadratic scaling of the ViT attention mechanism. However, these methods are not limited to digital-born PDFs with internal layout and metadata. On the other hand, more cost-efficient compartmentalized approach divides the conversion from image to text into several canonical phases which usually rely on the internal layout of the PDF file. Despite having higher throughput for lower cost, these approaches are not applicable to scanned PDF files. The compartmentalized approach generally consists of the following steps: image pre-processing, layout detection, text box extraction, character recognition, and text ordering.

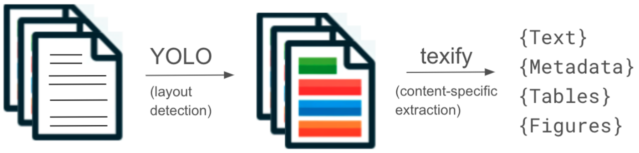

Figure 2: Parsing workflow for scientific PDFs. The neural optical recognition with eclectic output allows content-aware extraction of information from large-scale PDF collections.

We achieve across-the-board improvements in scaling performance in Oreo by combining the strengths of both end-to-end and compartmentalized approaches. PDFs are processed page by page and converted to images. Batches of pages are then parsed in a two-step process.

First, layout detection is performed using convolutional neural networks that identify regions corresponding to text, figures, and table categories. Text regions are further categorized into metadata (e.g., abstract, title, author, publication date) and content (e.g., paragraphs, footnotes). The YOLOv5 architecture used in our experiments provided the best trade-off between throughput, accuracy, and scalability.

Second, Texify is applied to image patches identified as text in the previous step to decode them into plain text. It achieves this by embedding the image using a Vision Transformer (ViT) and decoding the text autoregressively [59]. The decoded text items are subsequently recombined for each page and merged into the document's full text.

In addition to achieving a 4.5× speed-up over the current state-of-the-art approach, Marker, Oreo also surpasses both Nougat and Marker in multimodal capabilities. Specifically, Oreo can distinguish between 20 types of document assets, compared to Marker's three. Unlike Nougat, which can parse only text from PDFs, Oreo comprehensively handles a variety of document assets including figures, tables, equations, and code. Furthermore, Oreo explicitly parses metadata, which enables the storage of citations, author names, and references from scientific documents to facilitate digital archiving. Figure 2 depicts Oreo's overall document parsing workflow.

3.2 ColTrast: Query-Aware Encoder Finetuning

The discussion of the ColTrast query-aware encoder finetuning algorithm is organized as follows. Section 3.2.1 presents the design choices we made on the similarity metric, model size, and loss function along with our reasoning for each. Section 3.2.2 introduces the ColTrast training dataset and the procedure for curating it.

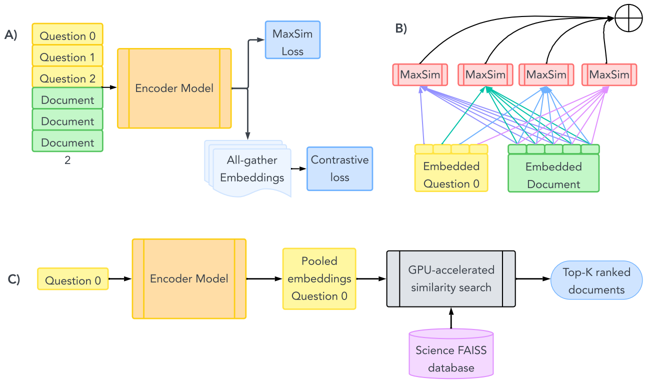

Figure 3: Encoder/retrieval models workflow using the ColTrast loss method. A) Training using ColTrast combines local token-level late-interaction loss with a contrastive loss operating on gathered embeddings. B) Details of late-iteration loss where each token in question/query is compared to each document. C) Inference time workflow. Each question is embedded, and then used in similarity search to extract relevant chunks from the scientific database.

3.2.1 *ColTrast Architecture and Implementation.* Embedding-based retrievers function on the premise that similar vectors indicate similarities in text. Therefore, in many question answering use cases, it is common to embed the query text and identify the most similar document chunk embeddings. The texts corresponding to these embeddings then are added as part of the in-context information the LLM can use to help answer the original question. In other words, the goal is to augment the LLM by providing text containing the information needed to answer the original question. The major determinants of retrieval performance are similarity metric and embedding performance, which is determined by model size, loss function, and training data.

We note that many of these performance determinants are double-edged: cutting back computation and improving scaling often comes at the cost of retrieval accuracy and vice-versa. However, we identified several places where we could improve accuracy with no or minimal loss of scaling performance, as described below.

Similarity metric: token-by-token vs. pooled embeddings. Although ColBERT is a performant retriever in terms of retrieval accuracy, its late interaction loss function involves a token-by-token comparison that scales as the product of the numbers of tokens in the document chunk and query. Figure 3 illustrates the detrimental effect of this quadratic process for a single comparison in terms of scaling performance. For a large-scale vector store with billions of chunks of text, these scaling barriers become impractical. Ideally, we would like to retain performance and scalability together. Scalabilty in this case depends on choice of similarity metric and we choose to replace the token-by-token similarity metric with the cosine similarity on the pooled embeddings. Normally, this would lead to performance degradation. We describe in the following how we overcame this problem by enhancing the accuracy of the embedding model.

Embedding performance: model size. For a well-trained model, model size is often a primary determinant of model performance. While BERT-style models such as ColBERT perform well on embedding tasks, especially for their size (∼110M parameters), billion parameter GPT-style models have started to surpass moderate BERT-style models on the embedding leaderboard. At the time of writing, a 7B parameter GPT-style model named SFR-Embedding-Mistral [54] has the best overall performance on embedding tasks, meaning it is the embedding model that most accurately captures semantic meaning appropriately. It is therefore likely that SFR-Embedding-Mistral is a good basis for enhancing the performance of our similarity metric that uses pooled embeddings.

Embedding performance: loss function. Fine-tuning on relevant tasks can enhance performance. However, increasing the model size by 70× limits our batch sizes and requires model sharding across GPUs, hurting our ability to fine-tune the embedding representation. To resolve these issues, we fine-tune the 8-bit version of SFR-Embedding-Mistral model using QLoRA [23] with a batch size of 24 per GPU. Since there is no need for model sharding, we can now use simple data parallel approaches (e.g., ZeRO-1) and are able to achieve linear scaling up to 400 nodes (1,600 GPUs) of the Polaris supercomputer (see Section 5 for details).

We use contrastive loss to improve our embedding models, which benefits from large batch sizes [12]. Naively increasing batch size by reducing loss from all ranks and applying contrastive or LI loss is limited by dense embeddings becoming large with increased world size, affecting LI more acutely than contrastive losses. We apply LI loss to the local rank only due to its memory and computational demands. For contrastive loss, pooled embeddings from all ranks are gathered, and loss is calculated with the local rank compared to $\min(N, W)$ samples, where N is the maximum to consider and W is the total samples across all ranks. The total loss per iteration is $L = (L_{LI} + L_C)/2$, where L_{LI} is maxsim loss [41] and L_C is contrastive loss [29]. This "ColTrast" loss method is illustrated in Figure 3. Following this fine-tuning algorithm produces an embedding representation that clusters related scientific concepts (discussed further in Section 4).

3.2.2 *ColTrast Finetuning Dataset.* We finetune ColTrast on three scientific paper datasets, totaling 82,750 papers, that we curated. These papers span diverse topics, including low-dose radiation and cancer biology, antimicrobial peptide literature, and SARS-CoV-2

Table 1: Composition of the ColTrast finetuning dataset with a further description provided in Section 3.2.2.

Domain	PDF Count	Chunk Count	Generated questions
Peptides	10,124	123,685	70,866
Cancer	18,034	208,803	114,535
COVID-19	54,592	519,412	270,493
Total	82,750	851,900	455,894

research. The high-level composition of the combined dataset is summarized in Table 1, and the sub-domain distribution of the antimicrobial peptide dataset is illustrated in Figure 4 for additional clarity on its diversity.

To enhance the effectiveness of our retrieval system, we apply a semantic chunking algorithm to the text of these papers. Semantic chunking is the process of dividing the content of a paper into concise, coherent segments or "chunks" that each encapsulate a complete semantic idea. This technique involves iterating over the encoded sentences using SFR-Embedding-Mistral [54] and continuously adding to a segment as long as the cosine similarity between consecutive sentences remains above a predetermined threshold. This method not only aids in maintaining the contextual integrity of the information but also optimizes the retrieval process by ensuring that each chunk represents a distinct concept or topic. Such organization of text before vectorization is widely recognized to improve the precision and relevance of retrieved documents, thereby advancing the state of the art in document parsing for LLM-based retrieval systems.

For each semantic chunk created, we then employ Mistral-7B-Instruct-v0.2 [37] to generate a high-level question that uses the chunk as a reference or resource for its answer. This process results in a collection of 455,894 question-chunk pairs, collectively referred to as the ColTrast fine-tuning dataset, as shown in Table 1. This structured approach not only improves the efficiency of our retrieval system but also significantly enhances the quality of generated questions, facilitating more accurate and contextually relevant responses.

3.3 Distributing HiPerRAG with HPC and Warmstart Optimization

We use the Parsl parallel programming library [11] to distribute the execution of our PDF parsing, semantic chunking, chunk encoding, and question sampling workflows. Parsl supports execution across diverse HPC resources and schedulers and has been shown to scale to hundreds of thousands of workers and thousands of tasks per second. However, standard Parsl assumes that tasks are pure functions, which is not conducive to persisting shared data structures, such as ML models, between tasks. For example, loading models for semantic chunking is an I/O-heavy operation and takes longer when more nodes read the same weight files concurrently: nearly nine minutes at worst, as shown in Table 2.

Thus, we implement a model registry that makes Parsl workers stateful actors that can persist a shared state across task invocations. The registry is implemented as a module-level global singleton variable that caches the return result of a Python function or class initialization in a dictionary using a hash of the arguments and keyword arguments as a unique key. The registry permits only one object at a time. Before a new object is created and registered, any existing object is automatically destroyed to free up shared resources, such as GPU memory [16]. In our use case, the registry captures a reference to the model upon its first initialization within a worker process (cold start), then returns the cached model for all subsequent task invocations (warm start). This method enhances system utilization by minimizing I/O overheads, amortizing the cost of model loading, and keeping the model in device memory (e.g., CUDA memory) to reduce memory transfers from host to device.

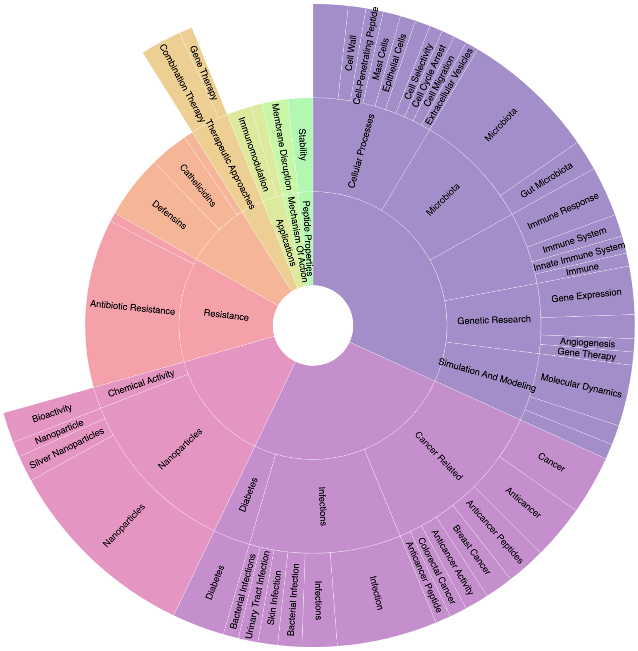

Figure 4: Sub-domain distribution of the antimicrobial peptide dataset for ColTrast finetuning.

Table 2: Semantic chunking model load times on Polaris and Sunspot increase at larger scales. The 2- and 128-node values correspond to Polaris, while the 4- and 96-node values correspond to Sunspot. All times in seconds.

	Nodes				
System	**2 / 4**	**32**	**64**	**96 / 128**	**256**
Polaris	36 ± 21	88 ± 28	114 ± 24	130 ± 23	362 ± 88
Sunspot	172 ± 73	218 ± 107	209 ± 160	535 ± 5	—

4 HiPerRAG Evaluation

It is important to reiterate that all RAG systems are made up of multiple independent components. Accordingly, HiPerRAG consists of the Oreo document parser, the ColTrast query-aware encoder, and an arbitrary generator model to produce a response in light of the relevant content retrieved. Our discussion in this section commences with an evaluation of Oreo in Section 4.1. The evaluation of ColTrast in retrieval accuracy will follow in Section 4.2. Section 4.3

will introduce the scientific Q/A and retrieval accuracy evaluation datasets that this work contributes. Section 4.4 will conclude with an evaluation of various HiPerRAG configurations on five scientific Q/A benchmarks, two of which are our contribution.

4.1 Oreo Document Parsing Evaluation

We evaluate Oreo, Marker, and Nougat on $n = 100$ scientific documents spanning eight domains: mathematics, physics, chemistry, biology, engineering, medicine, economics, and computer science, and six publishers: ArXiv, BioRxiv, MedRxiv, BMC, MDPI, and Nature. Ground truth text is sourced from the full-text HTML versions of the articles.

Table 3: Accuracy and throughput of image-based parsers on diverse scientific documents.

	Oreo (ours)	**Nougat**	**Marker**
Accuracy (BLEU) [%]	46.34	46.42	**56.90**
Accuracy (CAR) [%]	**73.92**	70.92	73.51
Throughput [PDFs/GPU sec]	**0.55**	0.12	0.09

Parsing quality is measured using bilingual evaluation understudy (BLEU) and character accuracy rate (CAR). BLEU evaluates n-gram overlap between parsed and ground truth text [58]. CAR measures character-level precision, particularly valuable for scientific equations and numerical data [83].

The results in Table 3 show that Marker achieves the highest BLEU score but has the lowest throughput. Nougat offers a competitive BLEU score but the lowest CAR. Notably, Oreo excels with the highest CAR and throughput while approximately matching Nougat's BLEU performance. Therefore, Oreo maximizes the number of accurately parsed tokens, making it a suitable option for large-scale document parsing.

4.2 ColTrast Retrieval Evaluation

To evaluate the ColTrast retrieval approach, we utilize two datasets: (1) our BioSynthQP dataset described in Section 4.3.3, and (2) a 5% held-out evaluation dataset from the ColTrast training set, as described in Section 3.2.2. Rather than sampling the evaluation set randomly, we ensure that the training and evaluation sets contain passage/query pairs from distinct scientific documents. Table 4 shows results of training with different model architectures and loss paradigms against the evaluation datasets. When provided with a question, the documents are ranked according to the cosine similarity between the question and document embeddings.

We report the precision as $P = N_{rel}/N_{ret}$ for the number of relevant retrieved documents, N_{rel}, and the total number of retrieved documents, N_{ret}. In the case of P@10, N_{ret} is set to 10. MRR@10 is calculated as $MRR = N_q^{-1} \sum_i \text{rank}_i^{-1}$ where rank$_i$ is the ranking of the positive paired document for the i^{th} query, N_q is the total number of queries, and @10 sets that if the positive document is ranked beyond 10, that term of MRR is zero. All models in the upper section of Table 4 were trained with the IA^3[50] parameter-efficient fine-tuning (PEFT) method, except ColTrast-M-Q-, since the QLoRA approach requires LoRA PEFT[35]. We performed ablations to examine the effectiveness of the ColTrast combined loss method. BERT-CL-S uses contrastive loss only, BERT-LI-S only has late-interaction loss, and <*Model name*>-ND-S denotes models with the ColTrast loss method but excluding the all-gather for the contrastive loss. The base model used for fine tuning is denoted by "M" for SFR-Mistral and "B" for BERT-base. We observe that performance on evaluation metrics is improved with the ColTrast loss method, as opposed to utilizing only LI or contrastive losses and that, for these models where a decent batch size is allowed, ND performs comparably to the distributed loss. The models in the lower half of the table are pre-existing approaches to the embedding and retrieval task included for comparison.

Table 4: Model performance on the BioSynthQP dataset and evaluation split of data. Abbreviations used include: BSQP - BioSynthQP dataset; Eval - the evaluation split; M - Salesforce Research/SFR-Mistral base model; B - BERT base model; Q - quantized low-rank approximation (QLoRA) parameter-efficient fine-tuning; ND - No distributed all-gather in contrastive loss; CL - contrastive loss only; LI - Late-Interaction Loss only; S - Model with ColTrast loss (CL + LI).

	P@10	MRR@10	MRR@20	Top-20
Name	BSQP	BSQP	Eval	Eval
ColTrast-M-Q-S	0.74	0.35	**0.93**	0.62
ColTrast-M-Q-ND-S	**0.81**	0.51	0.90	**0.98**
ColTrast-B-S	0.33	0.03	0.21	0.05
ColTrast-B-ND-S	0.72	0.46	0.84	0.96
BERT-CL-S	0.28	0.45	0.184	0.04
BERT-LI-S	0.03	0.003	0.06	0.003
PubMedBERT	0.72	**0.65**	0.22	0.39
ColBERT-v2	0.31	0.11	0.371	0.97
BERT-Base	0.31	0.03	0.21	0.37

4.3 Scientific Question-Answering and Retrieval Accuracy Evaluation Datasets

Question-answering (QA) datasets are pivotal for both training and evaluating LLMs in various scientific domains. These benchmarks are especially valuable for gauging a model's depth of domain-specific knowledge, its ability to understand intricate questions, and its proficiency in quantifying uncertainty during the generation of open-ended responses [42]. In Sections 4.3.2 and 4.3.1, we present two novel scientific QA datasets that we used to evaluate RAG configurations within the HiPerRAG framework. In Section 4.3.3 we present a synthetic retrieval accuracy evaluation dataset, which was used for assessing retrieval accuracy of query-aware scientific encoders, such as ColTrast that we present in this work.

4.3.1 ProteinInteractionQA. We compiled a dataset of 16,009 antimicrobial peptides from three sources: The Antimicrobial Peptide Database (APD) [77], the Database of Antimicrobial Activity and Structure of Peptides (DBAASP) [61], and the Database of Antimicrobial Resistance Peptides (DRAMP) [66]. For each peptide, we utilized the UniProt API [73] to retrieve data on proteins that interact with them. This effort identified 7591 peptides with known interactants. Utilizing the instruction-tuned Mistral7B model [36], we generated multiple-choice questions from this experimentally validated data. The resulting task, ProteinInteractionQA, poses questions such as "What protein does <peptide-name> interact with?" The correct answer includes all known interactants from UniProt, while the distractors comprise non-interactants randomly selected from other peptides in the dataset. It is worth noting that formatting our curated data into a Q/A task did not require a state-of-the-art LLM. Mistral7B proved sufficient for this purpose, thanks to its ability to follow instructions and produce structured output effectively.

4.3.2 ProteinFunctionQA. To curate the ProteinFunctionQA, we downloaded functional descriptions for 17,646 antimicrobial peptides from the UniProt database. We then employed the Mistral7B [36] instruction-tuned LLM to generate multiple-choice questions based on this experimentally annotated data. The correct answer reflected the ground-truth function of the peptide as retrieved from UniProt, while the distractors were incorrect functions sampled from other peptides within the dataset. The task requires selecting the correct function for a given peptide.

4.3.3 BioSynthQP. We created this synthetic biomedical data set by using GPT-4 prompt engineering with the express purpose of evaluating retrieval accuracy. BioSynthQP features scientific questions surrounding subdomains of medicine such as virology, oncology, and cardiology. For each subdomain, we generate a set of questions, along with 10 paragraphs for each question with a decreasing level of accuracy and relevance. In this context, the most relevant paragraph constitutes an answer that contains keywords, empirical results, and citations. The least relevant paragraph, on the other hand, digresses away from the question, lacks scientific basis, and citations, but still is broadly related to medicine. We apply human supervision to the postprocessing of the synthetic content to guarantee the accuracy of the quality labels assigned to each sample. Consequently, BioSynthQP presents a demanding retrieval task that evaluates an encoder's capability to generate embeddings that accurately capture quality characteristics within samples from the same domain.

4.4 Scientific Q/A Evaluations

We analyze the performance of our scientific Q/A from three angles: composition of the retrieval corpus, encoder model employed for retrieval, and the generator that leverages the retrieved content to answer questions. The two scientific corpora that we curated for this analysis differ greatly in terms of their domain composition and document size. As shown in Table 1, PLC consists of 10,124 PubMed articles about proteins. We encode this corpus with PubMedBERT, an encoder-based LLM specifically fine-tuned on PubMed articles [34]. On the other hand, Scientific Literature Corpus (SLC) contains over 3.6 million articles across numerous domains. We encode SLC with our Coltrast-B-S and Coltrast-M-Q-S encoders, which have been fine-tuned on a comprehensive scientific dataset (Section 4.2). We utilize instruct-finetuned versions of two generators Mistral-7B and Mixtral-8x7B, both of which are instruct-finetuned. Table 5 compares encoder models with respect to accuracy on five scientific Q/A tasks. SciQ [78] covers crowd-sourced science exam questions on Biology, Chemistry, and Physics. PubMedQA contains biomedical research questions. LitQA [44] also focuses on the biomedical domain, but features questions that can only be answered from knowledge in full-text papers. ProteinInteractionQA and ProteinFunctionQA are presented in this work (Section 4.3). We also provide baseline results where we do not use retrieval.

We observe that the Mixtral8x7B model which retrieves with PubmedBERT [33] outperforms GPT-4 (75.2% accuracy) on the PubmedQA benchmark [2] despite having access to a much smaller biomedical corpora (PLC), and having a smaller parameter count (14-billion parameters utilized at inference). Additionally, this model also outperforms PubmedGPT [15], a domain-specific LLM with 2.7 billion parameters pre-trained from scratch on biomedical data. This result suggests that retrieval-augmented generation at scale can improve the accuracy of LLMs in a data-efficient manner and render general-purpose LLMs more performant than domain-specific models on scientific QA tasks. Moreover the Mixtral8x7B model which uses the (ColTrast-B-S) encoder answers 90% of the questions in the comprehensive SciQ dataset. ColTrast-B-S retrieval and SLC corpora lead to a 12% improvement of this model on that benchmark. This result is consistent with our hypotheses that combining a novel metric learning-based retrieval strategy with the ability of retrieving from millions of scientific articles can equip LLMs with domain-specific knowledge beyond their training data.

Table 5: Comparison of different encoder models with Protein Literature Corpus (PLC) and Scientific Literature Corpus (SLC). Results are reported as accuracy (%) on multiple-choice scientific question-answering benchmarks. Abbreviations used include: PMB - PubMedBERT Encoder; ColTrast-B-S - ColTrast encoder based on BERT-base; ColTrast-M-Q-S - ColTrast encoder based on SFR-Mistral with QLoRA.

Corpus	**Generator (Encoder)**	**SciQ**	**PubmedQA**	**LitQA**	**Protein-InteractionQA**	**Protein-FunctionQA**
PLC	Mistral (PMB)	0.80	0.45	0.32	0.44	0.32
	Mixtral8x7B (PMB)	0.88	**0.76**	0.34	0.70	0.52
SLC	Mistral (ColTrast-B-S)	0.82	0.45	0.32	0.40	0.32
	Mistral (ColTrast-M-Q-S)	0.85	0.44	0.24	0.43	0.332
	Mixtral8x7B (ColTrast-B-S)	**0.90**	0.75	0.34	0.66	**0.526**
BASE	Mistral	0.88	0.59	**0.40**	0.44	0.36
	Mixtral8x7B	0.78	0.73	0.38	**0.70**	0.36

5 HiPerRAG Scaling and Performance

We conduct extensive scaling experiments on document parsing, semantic chunking, and encoder fine-tuning steps of HiPerRAG . In Section 5.1 we describe the HPC platforms on which we conducted our experiments. Section 5.2 will discuss our experiments and their results in detail.

5.1 Hardware Platforms Utilized

We evaluate the performance and scale on three diverse supercomputing systems: Polaris at the Argonne Leadership Computing Facility (ALCF), Sunspot at ALCF, and Frontier at the Oak Ridge Leadership Computing Facility (OLCF). In the November 2024 Top-500 list [71], Polaris is ranked #47 with a peak of 34 PFLOPS and

Table 6: GPU supercomputing systems used for evaluation.

	Polaris	Sunspot	Frontier
November-2024 Top 500#	47	–	2
System size (nodes)	560	128	9408
CPU	AMD Milan	Intel Sapphire Rapids	AMD
Sockets/node (total cores)	1 (32)	2 (104)	1 (64)
Node CPU Memory (TB)	0.5	1.0	0.5
GPU	NVIDIA A100	Intel Data Center Max (PVC)	AMD MI250X
Number of GPUs per node	4	6	4
GPU Memory (GB)	40	128	128
GPU Memory Technology	HBM2	HBM2e	HBM2e
GPU Memory BW (TB/s)	1.5	2.0	2.0
Interconnect	HPE SS-11	HPE SS-11	HPE SS-11
NICs per node	2	8	4
Network GB/s per direction	25	200	100
# Nodes (GPUs) scaled	450 (1800)	96 (576, 1152)	96 (384,768)

Frontier is ranked #2 with a peak of 2.055 EFLOPS. Table 6 compares the three systems used for evaluation.

Polaris is an HPE Apollo Gen10+ system with 560 nodes interconnected by an HPE Slingshot-11 network with a Dragonfly topology. Each node consists of an AMD "Milan" processor with 32 cores and 512GB of system memory. four 40 GB NVIDIA A100 GPUs, and two Slingshot-11 25 GB/s network adapters. Each NVIDIA A100 GPU is capable of achieving a peak of 19.5 TFLOPS in FP32, 156 TFLOPS in TF32, and 312 TFLOPS in FP16 and BF16.

The **Sunspot** Test and Development System (TDS) is a 128 node HPE Cray EX system. Each node consists of two Intel Sapphire Rapids Xeon CPU Max Series and six Intel Ponte Vecchio Data Center GPU Max Series, as has 1 TB memory Each Xeon CPU has 52 physical cores (two hardware threads per core). GPUs have 128 GB of high-bandwidth memory (HBM2e) memory and are connected within each node by an all-to-all interconnect. Each node has 8× HPE Slingshot-11 NICs, providing an injection bandwidth of 200 GB/s.

Frontier is an HPE Cray EX system comprised of 74 Olympus rack, 128-AMD-compute-node, HPE cabinets, for a total of 9408 AMD compute nodes. Each compute node comprises a single 64-core AMD CPU (two hardware threads per core), 512 GB of DDR4 memory, and four AMD MI250X GPUs. Each GPU has two Graphics Compute Dies (GCDs) for a total of 8 GCDs per node; GCDs have 64 GB of HBM2e. The nodes are connected with 4× HPE Slingshot-11 25 GB/s NICs providing a node-injection bandwidth of 100 GB/s.

5.2 Scaling the HiPerRAG Workflow

We trained the embedding retriever models with the Huggingface Accelerate library and the Huggingface trainer. Aside from implementation simplicity, the trainer also supplies estimates of total FLOPs and run time at each checkpoint. From these estimates, and a known world size, we estimated TFLOPS per GPU. The trainer (after a complete run) also supplies estimates of training steps per second. We use these metrics to derive the performance results presented in the next section.

Figure 5-A shows the performance of the Oreo and Nougat parsers as we strong scale on Polaris and Sunspot by increasing the number of accelerators to 1024 GPUs on Polaris (256 nodes) and to 1152 tiles on Sunspot (96 nodes, each with 6 GPUs and 12 tiles). On Polaris at 1024 accelerators, Oreo achieves 118.2 samples/s and Nougat 38.9 samples/s: a 3× improvement in throughput with Oreo over Nougat. This improved throughput is key to realizing our vision of being able to parse and extract, in a scalable manner with high quality, all of the scientific literature generated.

Next, we compare the performance of Nougat processing on Polaris and Sunspot. We observe a throughput of 35.62 samples/s on 1152 tiles (96 nodes) of Sunspot and 38.9 samples/s on 1024 GPUs (256 nodes) of Polaris. At a node-to-node level, we observe a 2.36× improvement in performance on Sunspot over Polaris. This result is expected as we have more GPUs and memory on each Sunspot node. At an accelerator level, comparing a single A100 GPU to a Intel Max GPU tile, we currently observe a 26% improvement for an A100 GPU in comparison to a tile on the Intel Max Data Center GPU. We attribute this primarily to the current state of optimization of the software stack here. In all cases, we observe a linear scaling in throughput as we scale out, which we attribute to the embarrassingly parallel nature of the parsing workflows.

For end-to-end parsing, Oreo achieves 40.7 TFLOPS/GPU peak with a sustained average of 37.2 TFLOPS/GPU on an NVIDIA A100, while Nougat achieves 0.43 TFLOPS/GPU peak and a sustained average of 0.28 TFLOPS/GPU. Thus, Oreo achieves 94.6× better compute performance than Nougat while sustaining a 4× improvement in throughput. This result is expected as Oreo follows a compartmentalization strategy that adds compute requirements in two ways. First, layout detection and combination of text items require tensor operations. Second, once text items are transcribed, they need to be mapped to their proper position. These innovations enable Oreo to process efficiently multiple file formats and multi-modal document assets including figures, tables, equations, and code.

We next compare the strong scaling performance of our end-to-end Semantic Chunking workflow on Polaris and Sunspot. As seen from Figure 5-B, we observe linear scaling, in terms of throughput in samples/s, as we scale with the number of accelerators. This result is expected given the embarrassingly parallel nature of the workflow. We observe a slight dip in performance at 256 accelerators (64 nodes), likely due to storage system limitations when reading the data to be chunked. We observe a throughput of 385.7 samples/s on 1024 accelerators (256 nodes) of Polaris. On Sunspot, we observe a throughput of 319.8 samples/s on 1152 accelerators (96 nodes). Normalizing these at an accelerator level, we observe a 35% improvement in achievable throughput on an A100 GPU in comparison to a tile of an Intel GPU. In terms of a node-to-node

Table 7: Estimated floating point operations and throughput for embedding/retriever model architectures during 100-step scaling runs using 400 GPUs on Polaris supercomputer (<*model*>-P). BERT-base-LoRA-F was run on 2400 GCDs (1200 GPU) on Frontier supercomputer. Throughput is in samples per second, total samples (S_T) in millions of samples.

Name	TFLOPS/ GPU	FLOPs (10^{18})	Samples/s	S_T (10^6)
Mistral-LoRA-P	0.71	2.40	152.2	0.64
Mistral-QLoRA-P	93.80	160.00	3496.4	3.84
BERT-base-LoRA-P	7.95	0.67	24,157.0	2.56
BERT-base-LoRA-F	1.92	3.10	8654.8	7.68

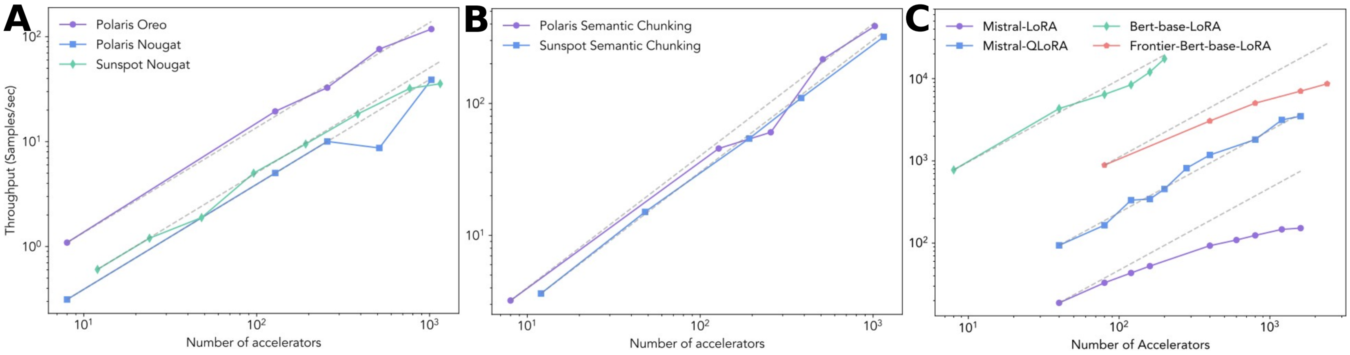

Figure 5: A) Strong scaling results for PDF parsing workflows on Polaris and Sunspot; B) Strong scaling results for semantic chunking workflows on Polaris and Sunspot; C) Strong scaling results for different model architectures on Polaris and Frontier. Unless noted, a run was accomplished on Polaris. Note that one GPU in this figure on Frontier corresponds to one GCD.

comparison, we achieve a 2.2× improvement on a Sunspot node in comparison to a Polaris node.

Figure 5-C shows strong scaling for training encoder models with samples/s as our throughput metric. Polaris runs include major model architectures used here: BERT base and mistral. Each model is trained using parameter-efficient fine-tuning—for computational efficiency and to help the model retain its knowledge from pretraining. Early experiments showed less than ideal scaling for Mistral-LoRA; since the model is too large for a single GPU, we sharded it across ranks by using Deepspeed ZeRO-3, and further employed parameter and optimizer state offloading in order to maximize batch size. To overcome this poor scaling, we adopted QLoRA training, where the model is first quantized to reduce memory pressure and then a LoRA adapter is applied as trainable parameters in full (bf16) precision. The Mistral-QLoRA approach yielded twofold benefits: first, the reduced memory footprint allowed us to load the full Mistral model into a single Nvidia-A100 with a batch size of 24; second, the reduced communication overhead compared to ZeRO-3 also enabled ideal scaling up to 400 nodes of Polaris.

Despite the success of QLoRA on Polaris, we were unable to employ a similar tactic on Frontier, as the BitsAndBytes quantization library required by transformers is not yet supported on MI250X. Performance in terms of TFLOPS/GPU, total FLOPs and sample throughput are enumerated for the scaling runs in Table 7. Notably, applying the QLoRA approach to SFR-Embedding-Mistral fine-tuning increases TFLOPS/GPU by 132×, and also achieves the highest TFOPS/GPU and total FLOPs of all measured models. Further, given the total document chunks of the SLC dataset (16.4 M samples), Mistral with QLoRA could iterate an epoch of data in 1.3 hours. Such throughput enables fine tuning on corpora of hundreds of millions of PDF documents.

6 Conclusions

We presented HiPerRAG as a scalable scientific knowledge synthesis framework that adapts existing LLMs with fast and efficient neural retrieval approaches to minimize hallucinations. Our methods enhance traditional RAG models by integrating contrastive learning and late interaction techniques to customize the distance metric based on the input query. This enables more accurate retrieval of relevant information from potentially millions of scientific documents and allows the model to effectively navigate a high-dimensional embedding space. By learning semantic connections among diverse domains, the system approaches state-of-the-art performance in scientific question answering benchmarks in a zero-shot manner (i.e., without the need for fine-tuning the generator). This is a significant advance for scientific literature data, where often, a model trained for a specific purpose may need extensive fine-tuning to be performant on other tasks.

Our work has implications for how knowledge from scientific literature is encoded and represented using LLMs. We have demonstrated a scalable workflow that enables a “plug and play” framework for any LLM of choice while enabling retrieval at scale. Semantic chunking allows us to learn viable contexts within scientific literature, while also pointing to (and potentially learning) new semantic interactions, which provide additional contexts for the models to organize emerging concepts or ‘trends’ in scientific research. We intend to use temporal evolution strategies (i.e., by predicting emerging concepts or research themes at a future date, given current scenarios) similar to approaches in [27, 49].

A further implication of our work is how knowledge graphs can be instantiated from these representations. The semantic chunking and retrieval frameworks are extensible, and can use any backend model; however, we believe that the implicit representations learned within the embeddings can be used to connect across concepts and scientific domains with rich annotations (that can be enabled by metric learning). These representations then can be used to build an ontology connecting various scientific concepts and disciplines embodied in recent work [25]. Moreover, it could be used to propose new hypotheses based on existing knowledge.

Extensions to HiPerRAG include the capability to pursue multimodality, especially in handling images, video, and audio along with text, which are particularly valuable for parsing scientific articles rich in tables and figures [19, 38, 82]. Although Oreo can output multimodal data, the lack of established multimodal benchmarks for scientific literature led us to focus on text retrieval as the primary evaluation method for HiPerRAG. Future work will experiment with multimodal LLMs as generators to incorporate images and tables that Oreo can parse from a PDF. We expect this to benefit newly emerging workflows—including agentic implementations [8, 62] that begin to power innovative scientific discovery workflows.

Acknowledgment

We thank the Argonne Leadership Computing Facility (ALCF) supported by the DOE under DE-AC02-06CH11357 at Argonne National Laboratory and the Oak Ridge Leadership Computing Facility (OLCF) at the Oak Ridge National Laboratory supported under DE-AC05-00OR22725. This project is supported by the Coalition for Epidemic Preparedness Innovations (CEPI) under the Disease X Program, the Argonne Laboratory Directed Research and Development (LDRD) program and US DOE LUCID: Low-Dose Understanding, Cellular Insights, and Molecular Discoveries.

References

[1] 2016. *Semantics, Analytics, Visualization. Enhancing Scholarly Data.* Springer International Publishing. https://doi.org/10.1007/978-3-319-53637-8

[2] 2023. PubMedQA: A Dataset for Biomedical Research Question Answering. PubMedQA Project Website. https://pubmedqa.github.io/ Accessed: 2023-10-11.

[3] 2024. PDFMiner. https://pypi.org/project/pdfminer/. Accessed: [12/7/2024].

[4] 2024. PyMuPDF Documentation. https://pymupdf.readthedocs.io/en/latest/. Accessed: [12/7/2024].

[5] 2024. PyPDF Documentation. https://pypdf.readthedocs.io/en/stable/. Accessed: [12/7/2024].

[6] Zeeshan Ahmed and Thomas Dandekar. 2017. MSL: Facilitating automatic and physical analysis of published scientific literature in PDF format. *F1000Research* 4 (April 2017), 1453. https://doi.org/10.12688/f1000research.7329.2

[7] Opeoluwa Akinseloyin, Xiaorui Jiang, and Vasile Palade. 2024. A question-answering framework for automated abstract screening using large language models. *Journal of the American Medical Informatics Association* 31, 9 (July 2024), 1939–1952. https://doi.org/10.1093/jamia/ocae166

[8] Zhiyu An, Xianzhong Ding, Yen-Chun Fu, Cheng-Chung Chu, Yan Li, and Wan Du. 2024. Golden-Retriever: High-Fidelity Agentic Retrieval Augmented Generation for Industrial Knowledge Base. arXiv:2408.00798 [cs.IR] https://arxiv.org/abs/2408.00798

[9] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511* (2023).

[10] Sören Auer, Dante A. C. Barone, Cassiano Bartz, Eduardo G. Cortes, Mohamad Yaser Jaradeh, Oliver Karras, Manolis Koubarakis, Dmitry Mouromtsev, Dmitrii Pliukhin, Daniil Radyush, Ivan Shilin, Markus Stocker, and Eleni Tsalapati. 2023. The SciQA Scientific Question Answering Benchmark for Scholarly Knowledge. *Scientific Reports* 13, 1 (May 2023). https://doi.org/10.1038/s41598-023-33607-z

[11] Yadu Babuji, Anna Woodard, Zhuozhao Li, Ben Clifford, Rohan Kumar, Lukasz Lacinski, Ryan Chard, Justin Wozniak, Ian Foster, Michael Wilde, Daniel Katz, and Kyle Chard. 2019. Parsl: Pervasive Parallel Programming in Python. In *ACM International Symposium on High-Performance Parallel and Distributed Computing*.

[12] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning Representations by Maximizing Mutual Information Across Views. *arXiv preprint arXiv:1906.00910* (2019).

[13] Sachin Banker, Promothesh Chatterjee, Himanshu Mishra, and Arul Mishra. 2024. Machine-assisted social psychology hypothesis generation. *American Psychologist* 79, 6 (Sept. 2024), 789–797. https://doi.org/10.1037/amp0001222

[14] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. 2023. Nougat: Neural optical understanding for academic documents. *arXiv preprint arXiv:2308.13418* (2023).

[15] E. Bolton, D. Hall, M. Yasunaga, T. Lee, C. Manning, and P. Liang. 2022. Stanford CRFM Introduces PubMedGPT 2.7B. Stanford HAI. https://hai.stanford.edu/news/stanford-crfm-introduces-pubmedgpt-27b

[16] Alexander Brace and J. Gregory Pauloski. 2023. https://github.com/braceal/parsl_object_registry. Accessed: 2024-10-09.

[17] Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2023. BooookScore: A systematic exploration of book-length summarization in the era of LLMs. https://doi.org/10.48550/ARXIV.2310.00785

[18] Kexin Chen, Junyou Li, Kunyi Wang, Yuyang Du, Jiahui Yu, Jiamin Lu, Lanqing Li, Jiezhong Qiu, Jianzhang Pan, Yi Huang, Qun Fang, Pheng Ann Heng, and Guangyong Chen. 2024. Chemist-X: Large Language Model-empowered Agent for Reaction Condition Recommendation in Chemical Synthesis. arXiv:2311.10776 [cs.IR] https://arxiv.org/abs/2311.10776

[19] Wenhu Chen, Hexiang Hu, Xi Chen, Pat Verga, and William W. Cohen. 2022. MuRAG: Multimodal Retrieval-Augmented Generator for Open Question Answering over Images and Text. arXiv:2210.02928 [cs.CL] https://arxiv.org/abs/2210.02928

[20] Johan SG Chu and James A Evans. 2021. Slowed canonical progress in large fields of science. *Proceedings of the National Academy of Sciences* 118, 41 (2021), e2021636118.

[21] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. *arXiv preprint arXiv:2309.03883* (2023).

[22] Steven Deitz and Christina Freyman. 2024. *Science and Engineering Indicators 2024: The State of U.S. Science and Engineering.* Technical Report NSB-2024-3. National Science Foundation, Alexandria, VA. https://ncses.nsf.gov/pubs/nsb20243

[23] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. *arXiv preprint arXiv:2305.14314* (2023).

[24] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The Faiss library. arXiv:2401.08281 [cs.LG]

[25] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. arXiv:2404.16130 [cs.CL] https://arxiv.org/abs/2404.16130

[26] Lutfi Eren Erdogan, Nicholas Lee, Siddharth Jha, Sehoon Kim, Ryan Tabrizi, Suhong Moon, Coleman Hooper, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2024. TinyAgent: Function Calling at the Edge. arXiv:2409.00608 [cs.CL] https://arxiv.org/abs/2409.00608

[27] Santo Fortunato, Carl T. Bergstrom, Katy Börner, James A. Evans, Dirk Helbing, Staša Milojević, Alexander M. Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, Alessandro Vespignani, Ludo Waltman, Dashun Wang, and Albert-László Barabási. 2018. Science of science. *Science* 359, 6379 (2018), eaao0185. https://doi.org/10.1126/science.aao0185 arXiv:https://www.science.org/doi/pdf/10.1126/science.aao0185

[28] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. arXiv:2101.00027 [cs.CL] https://arxiv.org/abs/2101.00027

[29] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Conference on Empirical Methods in Natural Language Processing*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 6894–6910. https://doi.org/10.18653/v1/2021.emnlp-main.552

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

[31] Aditi Godbole, Jabin Geevarghese George, and Smita Shandilya. 2024. Leveraging Long-Context Large Language Models for Multi-Document Understanding and Summarization in Enterprise Applications. https://doi.org/10.48550/ARXIV.2409.18454

[32] Koustava Goswami, Lukas Lange, Jun Araki, and Heike Adel. 2023. SwitchPrompt: Learning domain-specific gated soft prompts for classification in low-resource domains. *arXiv preprint arXiv:2302.06868* (2023).

[33] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Transactions on Computing for Healthcare* 3, 1 (Oct. 2021), 1–23. https://doi.org/10.1145/3458754

[34] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Transactions on Computing for Healthcare* 3, 1 (Oct. 2021), 1–23. https://doi.org/10.1145/3458754

[35] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

[36] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[37] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel,

Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv e-prints*, Article arXiv:2310.06825 (Oct. 2023), arXiv:2310.06825 pages. https://doi.org/10.48550/arXiv.2310.06825 arXiv:2310.06825 [cs.CL]

[38] Pankaj Joshi, Aditya Gupta, Pankaj Kumar, and Manas Sisodia. 2024. Robust Multi Model RAG Pipeline For Documents Containing Text, Table & Images. In *3rd International Conference on Applied Artificial Intelligence and Computing*. 993–999. https://doi.org/10.1109/ICAAIC60222.2024.10574972

[39] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. arXiv:2004.04906 [cs.CL] https://arxiv.org/abs/2004.04906

[40] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4171–4186.

[41] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 39–48.

[42] Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTaiL: A textual entailment dataset from science question answering. In *AAAI Conference on Artificial Intelligence*, Vol. 32.

[43] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. OCR-free document understanding transformer. In *European Conference on Computer Vision*. Springer, 498–517.

[44] Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. 2023. PaperQA: Retrieval-augmented generative agent for scientific research. *Preprint ArXiv:2312.07559* (2023).

[45] Esther Landhuis. 2016. Scientific literature: Information overload. *Nature* 535 (07 2016), 457–458. https://doi.org/10.1038/nj7612-457a

[46] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. 2020. Contrastive Representation Learning: A Framework and Review. *CoRR* abs/2010.05113 (2020). arXiv:2010.05113 https://arxiv.org/abs/2010.05113

[47] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (Sept. 2019), 1234–1240. https://doi.org/10.1093/bioinformatics/btz682

[48] 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. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS'20)*. Curran Associates Inc., Red Hook, NY, USA, Article 793, 16 pages.

[49] Yiling Lin, James A. Evans, and Lingfei Wu. 2022. New directions in science emerge from disconnection and discord. *Journal of Informetrics* 16, 1 (2022), 101234. https://doi.org/10.1016/j.joi.2021.101234

[50] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning. *arXiv e-prints*, Article arXiv:2205.05638 (May 2022), arXiv:2205.05638 pages. https://doi.org/10.48550/arXiv.2205.05638 arXiv:2205.05638 [cs.LG]

[51] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. Pretrained Transformers as Universal Computation Engines. arXiv:2103.05247 [cs.LG] https://arxiv.org/abs/2103.05247

[52] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, and Xueqi Cheng. 2022. Pre-train a Discriminative Text Encoder for Dense Retrieval via Contrastive Span Prediction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 848–858. https://doi.org/10.1145/3477495.3531772

[53] Nicholas Matsumoto, Jay Moran, Hyunjun Choi, Miguel E Hernandez, Mythreye Venkatesan, Paul Wang, and Jason H Moore. 2024. KRAGEN: a knowledge graph-enhanced RAG framework for biomedical problem solving using large language models. *Bioinformatics* 40, 6 (June 2024). https://doi.org/10.1093/bioinformatics/btae353

[54] Rui Meng, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. SFR-Embedding-Mistral:Enhance Text Retrieval with Transfer Learning. Salesforce AI Research Blog. https://blog.salesforceairesearch.com/sfr-embedded-mistral/ Accessed: Apr 7, 2024.

[55] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL] https://arxiv.org/abs/1301.3781

[56] Radeen Mostafa, Mirza Nihal Baig, Mashaekh Tausif Ehsan, and Jakir Hasan. 2024. G-RAG: Knowledge Expansion in Material Science. https://doi.org/10.48550/ARXIV.2411.14592

[57] Mathias Wullum Nielsen and Jens Peter Andersen. 2021. Global citation inequality is on the rise. *Proceedings of the National Academy of Sciences* 118, 7 (2021), e2012208118. https://doi.org/10.1073/pnas.2012208118 arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.2012208118

[58] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[59] Vik Paruchuri. [n. d.]. Texify: Tool for Converting Text to LaTeX. https://github.com/VikParuchuri/texify. Accessed: [12/7/2024].

[60] Vik Paruchuri. 2024. Marker. https://github.com/VikParuchuri/marker. Accessed: [12/7/2024].

[61] Malak Pirtskhalava, Anthony A Amstrong, Maia Grigolava, Mindia Chubinidze, Evgenia Alimbarashvili, Boris Vishnepolsky, Andrei Gabrielian, Alex Rosenthal, Darrell E Hurt, and Michael Tartakovsky. 2021. DBAASP v3: Database of antimicrobial/cytotoxic activity and structure of peptides as a resource for development of new therapeutics. *Nucleic Acids Research* 49, D1 (2021), D288–D297.

[62] Chidaksh Ravuru, Sagar Srinivas Sakhinana, and Venkataramana Runkana. 2024. Agentic Retrieval-Augmented Generation for Time Series Analysis. arXiv:2408.14484 [cs.AI] https://arxiv.org/abs/2408.14484

[63] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*. 779–788.

[64] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (Eds.). Association for Computational Linguistics, Seattle, United States, 3715–3734. https://doi.org/10.18653/v1/2022.naacl-main.272

[65] Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. 2019. Molecular Transformer: A Model for Uncertainty-Calibrated Chemical Reaction Prediction. *ACS Central Science* 5, 9 (Aug. 2019), 1572–1583. https://doi.org/10.1021/acscentsci.9b00576

[66] Guobang Shi, Xinyue Kang, Fanyi Dong, Yanchao Liu, Ning Zhu, Yuxuan Hu, Hanmei Xu, Xingzhen Lao, and Heng Zheng. 2022. DRAMP 3.0: An enhanced comprehensive data repository of antimicrobial peptides. *Nucleic acids research* 50, D1 (2022), D488–D496.

[67] Dalmeet Singh Chawla. 2022. Massive open index of scholarly papers launches. *Nature* (Jan. 2022). https://doi.org/10.1038/d41586-022-00138-y

[68] Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint* (2024). https://arxiv.org/abs/2402.00159

[69] Open Source. [n. d.]. The Chroma Vector Database. https://docs.trychroma.com/

[70] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085* (2022).

[71] Top500. 2024. *November 2024 TOP500*. https://www.top500.org/lists/top500/2024/11/

[72] Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga Kononova, Kristin A. Persson, Gerbrand Ceder, and Anubhav Jain. 2019. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature* 571, 7763 (July 2019), 95–98. https://doi.org/10.1038/s41586-019-1335-8

[73] UniProt Consortium. 2021. UniProt: The universal protein knowledgebase in 2021. *Nucleic Acids Research* 49, D1 (2021), D480–D489.

[74] Richard Van Noorden. 2014. Scientists may be reaching a peak in reading habits. *Nature* (02 2014). https://doi.org/10.1038/nature.2014.14658

[75] Archit Vasan, Ozan Gokdemir, Alexander Brace, Arvind Ramanathan, Thomas Brettin, Rick Stevens, and Venkatram Vishwanath. 2024. High Performance Binding Affinity Prediction with a Transformer-Based Surrogate Model. In *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 571–580. https://doi.org/10.1109/IPDPSW63119.2024.00114

[76] Chengrui Wang, Qingqing Long, Meng Xiao, Xunxin Cai, Chengjun Wu, Zhen Meng, Xuezhi Wang, and Yuanchun Zhou. 2024. BioRAG: A RAG-LLM Framework for Biological Question Reasoning. arXiv:2408.01107 [cs.CL] https://arxiv.org/abs/2408.01107

[77] Zhe Wang and Guangshun Wang. 2004. APD: The antimicrobial peptide database. *Nucleic acids research* 32, suppl_1 (2004), D590–D592.

[78] Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing Multiple Choice Science Questions. *ArXiv* abs/1707.06209 (2017). https://api.

semanticscholar.org/CorpusID:1553193

[79] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. arXiv:2308.08155 [cs.AI] https://arxiv.org/abs/2308.08155

[80] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-training of text and layout for document image understanding. In *26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1192–1200.

[81] Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. 2021. FILIP: Fine-grained Interactive Language-Image Pre-Training. arXiv:2111.07783 [cs.CV] https://arxiv.org/abs/2111.07783

[82] Ruochen Zhao, Hailin Chen, Weishi Wang, Fangkai Jiao, Xuan Long Do, Chengwei Qin, Bosheng Ding, Xiaobao Guo, Minzhi Li, Xingxuan Li, and Shafiq Joty. 2023. Retrieving Multimodal Information for Augmented Generation: A Survey. arXiv:2303.10868 [cs.CL] https://arxiv.org/abs/2303.10868

[83] Xiaoqing Zheng, Haoyuan Peng, Yi Chen, Pengjing Zhang, and Wenqiang Zhang. 2015. Character-based parsing with convolutional neural network. In *Twenty-Fourth International Joint Conference on Artificial Intelligence.*

[84] Xiaochi Zhou, Shaocong Zhang, Mehal Agarwal, Jethro Akroyd, Sebastian Mosbach, and Markus Kraft. 2023. Marie and BERT–A Knowledge Graph Embedding Based Question Answering System for Chemistry. *ACS Omega* 8, 36 (Aug. 2023), 33039–33057. https://doi.org/10.1021/acsomega.3c05114