

**LEVERAGING LARGE-LANGUAGE MODELS AS COLLABORATIVE REASONING PARTNERS
TO ENHANCE SCIENTIFIC WORKFLOW**

by

DOUGLAS B. CRAIG

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2025

MAJOR: COMPUTER SCIENCE (Bioinformatics
& Computational Biology)

Approved By:

Advisor

Date

**© COPYRIGHT BY
DOUGLAS B. CRAIG
2025
All Rights Reserved**

DEDICATION

For my parents who laid the foundation
and my wife who was there as each block was added

è come la fabbrica del Duomo

ACKNOWLEDGEMENTS

First, thanks to my advisor Dr. Sorin Drăghici for his guidance and criticism during the course of this research. Thanks also to my committee members: Dr. Suzan Arslanturk, Dr. Weisong Shi and especially Dr. Alan Dombkowski who has supported my journey into the biological sciences from the start.

Thank you to members past and present of the ISBL team, especially: Minh Nguyen, Nicholas Adzibolosu, Muiz Khan, Dr. Samer Hanoudi and Dr. Azam Peyvandipour.

My heartfelt thanks to Dr. Judy Abrams for encouraging me to try *one more time* to complete this PhD, and to Dr. Greg Dyson, Dr. Wei Chen and Dr. Lance Heilbrun of the Bioinformatics & Biostatistics Core for their support as I began this journey.

Many thanks to Mike Catanzaro, Gabe Angelini-Knoll, Aaron Willcock and Clayton Hayes of WSU for developing, updating and maintaining the L^AT_EX template for dissertations on which this document is based.

Finally, love, as always, to my parents for their never ending encouragement, curiosity and support. To the home team: Duncan, Maddox, Simba, Ranger, Yogi, Harely and Red who have been with me over hill and dale during this journey. And, to my wife, Freida, who granted me the time and funds to support this research. Without her patience this would have been nothing more than yet another attempt.

TABLE OF CONTENTS

Dedication	ii
Acknowledgements	iii
List of Tables	xiii
List of Figures	xiv
List of Definitions	xix
List of Abbreviations	xxiv
Chapter 1 Introduction	1
1.1 Statement of the Problem	2
1.2 Research Objectives	3
1.2.1 LLM as Computational Component	3
1.2.2 Explainability and Reliability	3
1.2.3 Dynamic Information & Reasoning	4
1.2.4 Collaborative Note-taking	4
1.3 Organization	4
1.4 Disclaimer	6
Chapter 2 Background and Related Work	7
2.1 Large Language Models (LLM)	7
2.2 Retrieval-augmented Generation (RAG)	10
2.3 Agents, Reasoning and Planning	13
2.4 Scientific Discovery and Biology	16
Chapter 3 Research Design & Methodology	20
3.1 Research Design	20

3.1.1	LmRaC Command Line	21
3.1.2	LmRaC Web Application	23
3.1.3	LmRaC Collaborative Notes	24
3.1.4	LmRaC Future Work	26
3.2	Research Problem Domain	26
3.3	Implementation Details	29
Chapter 4	LmRaC Command Line	33
4.1	Abstract	33
4.2	Introduction	34
4.3	Related Work	35
4.4	Methodology	37
4.4.1	Domain Specific Question and Answer	37
4.4.2	Question and Answer with Experimental Context	42
4.4.3	Functional Interrogation of Experimental Results	42
4.4.4	Software Implementation	43
4.5	Experiments	44
4.5.1	Domain Knowledge	45
4.5.2	General Disease Questions	46
4.5.3	Experiment Specific Questions	48
4.5.4	Experimental Questions Using Saved Context	49
4.6	Discussion	50
4.6.1	Usefulness Evaluation	50
4.6.2	Metadata, Usefulness and Embeddings	51

4.6.3	Comparison to LLM Models	53
4.6.4	RAG Primacy Evaluation	55
4.7	Conclusion	56
Chapter 5	LmRaC Web Application	58
5.1	Abstract	58
5.1.1	Motivation	58
5.1.2	Results	58
5.1.3	Availability and implementation	59
5.2	Introduction	59
5.3	Materials and methods	61
5.3.1	Domain Specific Knowledge Bases	61
5.3.2	Sub-Question Generation	62
5.3.3	Vector Database Query and Paragraph Usefulness Assessment	62
5.3.4	Answer Generation and Quality Assessment	62
5.3.5	Question and Answer with Experimental Context	63
5.3.6	Quantitative Experimental Results	64
5.3.7	Software Implementation	64
5.4	Results	64
5.4.1	Creating Indexes by Asking Questions	65
5.4.2	Experimental Context and Data-aware Questions	67
5.4.3	Complex Questions	70
5.5	Conclusions	70
5.6	LmRaC Usage	71

5.7 Prerequisites	72
5.7.1 Docker	72
5.7.2 OpenAI	72
5.7.3 Pinecone	73
5.8 Quick Start	73
5.8.1 Next steps	81
5.9 Installation	81
5.9.1 Docker Engine (Linux)	81
5.9.2 Docker Desktop (Linux / Mac / Windows)	82
5.10 LmRaC Homepage	87
5.10.1 Commands	88
5.11 Indexes Window	90
5.11.1 Uploading pre-built indexes	91
5.12 Experiments Window	92
5.13 Functions Window	93
5.14 Answers Window	94
5.14.1 Saving Answers to Experiments	96
5.15 Configuration	97
5.15.1 Markdown Viewing	98
5.16 Usage: Q & A	98
5.16.1 Setting an Index	99
5.16.2 Asking a Question	99
5.16.3 Providing PubMed Sources	100

5.16.4 Tips	100
5.17 Usage - Experiments	101
5.17.1 Experimental Results	101
5.17.2 Creating an experimental context	102
5.17.3 Putting it all together	104
5.18 Usage - User-Defined Functions	106
5.18.1 Quick Start	106
5.18.2 Building your own REST API Server	107
5.18.3 Adding Functions (Server - REST Server)	108
5.18.4 Using Functions (Client - LmRaC)	111
5.18.5 General Functions	114
5.19 Indexes and Experiments and Functions	115
5.20 Troubleshooting	115
5.21 How To Cite	117
5.22 Contact	117
Chapter 6 LmRaC Collaborative Notes	118
6.1 Abstract	118
6.1.1 Keywords	119
6.2 Introduction	119
6.3 Methodology & Design	120
6.3.1 System Workflow	121
6.3.2 Case-based Reasoning	122
6.3.3 The Collaboration Process	122

6.3.4 Language Enabled Tools	123
6.3.5 Implementation	125
6.3.6 Executive Execution and Interface	125
6.4 Results	130
6.4.1 System Functionality and Usability	130
6.4.2 Innovations and Contributions	130
6.4.3 Qualitative Results	132
6.5 Discussion	133
6.6 Future Work	134
6.6.1 Cases	134
6.6.2 Tools	135
Chapter 7 Scientific Discovery	138
7.1 Abstract	138
7.2 Introduction	138
7.3 Methods & Design	140
7.3.1 System Workflow	140
7.3.2 Case-based Reasoning	141
7.3.3 The Collaboration Process	141
7.3.4 Language Enabled Tools	141
7.3.5 Implementation	142
7.4 Case Study	142
7.4.1 Research Question	142
7.4.2 Initiating Case-based Reasoning	144

7.4.3 Defining Scope	144
7.4.4 Refining the Problem	147
7.4.5 Quick Review	148
7.4.6 Literature Review	150
7.4.7 Knowledge Gaps	152
7.4.8 Database Review	153
7.4.9 Hypothesis Formulation	154
7.4.10 Experiments and Data Collection	156
7.4.11 Differential Expression Analysis	158
7.4.12 Experimental Insights and Reflection	159
7.5 Discussion	160
7.6 Conclusion	163
Chapter 8 Discussion	164
8.1 Results	164
8.1.1 Performance	164
8.1.2 Unexpected Findings	165
8.1.3 Contextualization	166
8.2 Strengths and Limitations of the Program	166
8.2.1 Unique Capabilities and Strengths	166
8.2.2 Limitations	167
8.3 Technical Challenges and Solutions	168
8.3.1 Challenges Encountered	168
8.3.2 Solutions and Trade-offs	169

8.4 Validation and Evaluation	169
8.4.1 Reliability and Accuracy	169
8.4.2 Testing Methods and Outcomes	170
8.4.3 Benchmark Testing	170
8.4.4 Case Studies	170
8.5 Broader Implications	171
8.6 Ethical and Social Implications	172
8.6.1 Ethical Dilemmas and Risks	172
8.6.2 Mitigations	172
8.7 Comparison to Existing Work	173
8.7.1 Advancing the State of the Art	173
8.7.2 Key Distinctions	173
8.7.3 Limitations of Comparison	173
Chapter 9 Conclusion	175
9.1 Innovation	175
9.2 Practical Significance	176
9.3 Interdisciplinary Approach	176
9.4 Ethical Computing	177
9.5 Foundation for Future Work	178
Chapter 10 Future Research	179
10.1 Just in Cases	179
10.1.1 Multi-faceted Similarity	179
10.1.2 Weighted Semantic Graphs	180

10.1.3 Discovery and Synthesis	180
10.1.4 Review and Reflection	181
10.2 Visualization as a Tool	182
10.3 Agent Extensions	183
10.3.1 Agentic Independence	183
10.3.2 Agentic Personalities	184
10.3.3 Agentic Attention, Frustration and Values	185
10.4 Creative Hallucination or Maybe Timothy Leary was Just Ahead of His Time	186
Appendix A Controlled Vocabularies	188
Appendix B Test Questions	190
Appendix C Sample Comparison Results	191
Appendix D Example Answer	211
Appendix E Functional Interface	217
References	219
Index	248
Abstract	251
Autobiographical Statement	253

LIST OF TABLES

Table 1	Gene vocabulary file. Includes: gene symbol, name, Entrez identifier, Ensembl identifier and Uniprot protein identifier.	188
Table 2	Disease vocabulary file. Includes: disease MeSH identifier and disease name.	189
Table 3	Pathway vocabulary file. Includes: pathway identifier (hsa=human species), pathway name and primary pathway references (PubMed PMID identifiers).	189
Table 4	Questions for testing and pairwise performance comparison between LLM models.	190

LIST OF FIGURES

Figure 1	The Transformer model architecture (Figure 1 from [1]). The left side represents the encoder. The right side represents the decoder.	8
Figure 2	A timeline of LLMs larger than 10B parameters (Fig. 3 from Zhao et al. [2])	9
Figure 3	The different steps and components of an LLM when being built (Fig. 25 from Minaee et al. [3]).	11
Figure 4	LmRaC research roadmap.	21
Figure 5	LmRaC architectures. (A) open-loop question and answer using domain specific information only (RAG_{dom}). (B) close-loop question and answer with experiment specific information (RAG_{exp}). (C) functional interrogation of experimental results (RAG_{fun}).	37
Figure 6	LmRaC base architecture. (A) high level open-loop question and answer, (B) detailed component implementation.	38
Figure 7	Differential gene expression results for the Chemokine Signaling Pathway. Exploded view shows changes in the expression of three genes: AC (ADCY5) down-regulated (blue); $G\alpha i$ (GNAI1) down-regulated (light blue); and, $G\beta\gamma$ (GNG4) up-regulated (light red).	45
Figure 8	Pairwise comparison of LmRaC with and without filtering for usefulness. Each point corresponds to one question (size=question complexity). Quality assessed by GPT4 on a scale of 1 to 10. Note: jitter added to differentiate overlapping points.	51
Figure 9	Correlation between usefulness score and embedding similarity. (A) paragraphs chosen based on metadata only. (B) paragraphs chosen based on metadata and embedding similarity.	52
Figure 10	Embedding scores of cited useful paragraphs. (A) logistic regression for useful paragraph citations. (B) distribution of embedding scores for cited (1) versus not cited (0) useful paragraphs.	53
Figure 11	Pairwise comparison of LLM model answer quality. Each point corresponds to one question (size=question complexity). Quality assessed by GPT4 on a scale of 1 to 10. Note: "hybrid" = LmRaC.	54

Figure 12	LmRaC workflow overview. (1) A user asks a question through the web application interface. This initiates a search of PubMed for articles relevant to mentioned genes, pathways and/or diseases. These articles are saved as domain knowledge. (2) Quantitative data needed to answer the question is retrieved through a user-defined REST API as part of the experimental context. (3) Answers from previous questions may be saved to extend and refine the experimental context. (4) LmRaC answers the question using available domain knowledge and experimental context.	65
Figure 13	LmRaC splash icon.	71
Figure 14	LmRaC web application homepage.	75
Figure 15	LmRaC pre-populating of an index.	77
Figure 16	LmRaC pulling the latest Docker image.	83
Figure 17	LmRaC running container from Docker image.	83
Figure 18	LmRaC Docker container settings.	84
Figure 19	LmRaC container launch.	86
Figure 20	LmRaC web application homepage.	87
Figure 21	LmRaC indexes window.	91
Figure 22	LmRaC uploading pre-built indexes.	92
Figure 23	LmRaC experiments window.	93
Figure 24	LmRaC functions window.	94
Figure 25	LmRaC answers window.	95
Figure 26	LmRaC answers save dialog in the answers window.	96
Figure 27	Markdown Viewer browser extension setup.	98
Figure 28	LmRaC error during function load.	114

Figure 29	System workflow. The user interacts with notes and system console. Notes may identify a problem for which the System Executive will search for previous case instances. Cases are themselves notes that utilize tools to access external functions and data. A case is then reused or adapted to the current problem.	121
Figure 30	Bootstrapping and progressive problem solving. A case may initially only include a starting point. From this the user, in collaboration with the LLM, can incrementally build a solution to a problem. Cases may also use this approach to progressively solve problems with user input/feedback.	123
Figure 31	Tools selection. In posing a question a case is retrieved. This case utilizes tools either explicitly, or by search based on a description of the problem and data for which a tool is suited. Tools are applied stepwise to effect a solution for a given case.	124
Figure 32	All LmRaC notes include front matter (stored as YAML in the markdown file). This metadata identifies the note as a LmRaC note, its unique identifier and current ENQ/ACK state.	126
Figure 33	The LmRaC console application provides the main user interface outside of Obsidian. It provides debug feedback as well as meta-control buttons for: pause, resume, retry and debug as well as user text and voice input.	127
Figure 34	CBR cases are stored as Obsidian notes.	129
Figure 35	Tool specifications are stored as Obsidian notes.	129
Figure 36	When a question is asked within a note the <i>Note Change</i> case is used to assess the type of question. A new instance of <i>Note Change</i> is created and referenced from within the user note using internal link markdown.	131
Figure 37	System workflow.	140
Figure 38	Defining the question type and identifying CBR cases (e.g., Mechanistic and Causal questions)	143
Figure 39	Instantiate intial CBR case, Mechanistic Gene Function, based on previous experience. Link this (red arrow) to the top-level user note.	144
Figure 40	Case reasoning step requiring user input as part of the scope definition action.	145

Figure 41	Requested user information for scope definition. User interleaves answers and then checks boxes. User may also use strike-through to indicate information that should be ignored (i.e., is out of scope) . . .	146
Figure 42	Final system composed question scope definition.	147
Figure 43	Refined problem statement and key questions. An additional question has been added by the user (blue highlight).	148
Figure 44	Explicit tool call to ask OpenAI o1 model for "quick" unauthoritative answer.	149
Figure 45	"Quick" answers to the refined problem's key questions.	149
Figure 46	Initiate a literature search of PubMed on the list of Key Questions. Search results are stored in <i>res1</i> and references for search results are passed to another tool to be summarized.	150
Figure 47	Literature Review answer to first Key Question. Answer is based <i>only</i> on PubMed articles with paragraph-level links to citations and bibliography [4–11].	151
Figure 48	GPT API is used to assess knowledge gaps. An explicit prompt is provided along with results from the previous Quick Review and Literature Review. The response is structured using a function prototype parameter.	152
Figure 49	First two identified knowledge gaps (of nine) summarized and annotated by difficulty and reward.	153
Figure 50	Step definition to use two external data sources (Wikipedia and GeneCards) to summarize more information about the TLR4 gene.	154
Figure 51	Using the results of multiple previous steps: formulate research hypotheses.	155
Figure 52	Proposed Hypothesis 2, including: rationale and potential stepwise approach.	155
Figure 53	<i>Implicit</i> search of external resources for datasets suitable for preliminary results.	156
Figure 54	The CBR case includes Suggested Resources. This includes one for gene expression (GEO) and one for microbiome profiling (SRA). . . .	157

Figure 55	Excerpts from the search using OpenAI o1 model. Public datasets are identified (red arrows). Searches may also be done using tools to directly access resource APIs.	158
Figure 56	<i>Explicit</i> tool identification for computing differential gene expression on retrieved datasets.	159
Figure 57	Preliminary results analysis and assessment of next steps sensitive to this analysis (if results are available).	160
Figure 58	First two (of six) suggested next steps for the investigation including, hypothesis refinement, validation and prospective controlled studies. .	161

LIST OF DEFINITIONS

agent In artificial intelligence, an autonomous entity that perceives its environment and takes actions to achieve specific goals, often used in systems designed to solve complex tasks or interact with other agents and humans.

artificial intelligence The field of computer science focused on creating systems capable of performing tasks that typically require human intelligence, such as reasoning, learning, problem-solving, and understanding language.

autoencoder A neural network architecture typically used for unsupervised learning whereby a dimensionally reduced representational encoding of the input data is learned through iterative regeneration of the input from that encoding.

back-propagation A core neural network algorithm that adjusts network weights by calculating gradients of the error with respect to each weight and propagating these gradients backward from the output layer to minimize overall error.

case-based reasoning (CBR) A problem-solving approach where solutions to new problems are derived by adapting solutions from similar past cases stored in a case library.

chain-of-thought (CoT) A reasoning technique where a model generates intermediate steps of reasoning or explanations before arriving at a final answer, improving the model's ability to handle complex tasks.

controlled vocabulary A predefined set of terms used to ensure consistent categorization and retrieval of information within a specific domain, often employed in databases and knowledge management systems.

deep learning A category of machine learning that relies on artificial neural network architectures. Examples include: convolutional neural networks, deep neural networks, generative models and autoencoders.

deep neural network An artificial neural network with multiple layers between input and output.

differential gene expression The process of comparing gene expression levels between different conditions, such as diseased vs. healthy tissues, to identify genes that are upregulated or downregulated in response to those conditions.

diffusion model A generative model that learns data distributions by simulating a process where data points are gradually corrupted (diffused) with noise and then reversing this process to generate new samples.

embedding vector A numerical representation of words or phrases in a continuous vector space, where semantically similar words are mapped to nearby points, facilitating efficient computation and comparison in machine learning tasks.

explainable AI (XAI) A field of AI focused on creating systems whose decisions and underlying reasoning can be easily understood and interpreted by humans.

generative adversarial network (GAN) A deep learning architecture composed of two neural networks arranged so that one network generates candidates while the other discriminates between generated data and true data. This contest proceeds with the first trying to “fool” the second thereby learning to produce improved candidates.

generative artificial intelligence Systems that can create new content, such as text, images, or audio, by learning patterns from existing data and generating outputs that resemble human-created content.

generative pre-trained transformer (GPT) A type of large language model designed for generating and understanding natural language from pre-training on diverse text data and fine-tuning for specific tasks.

gene set enrichment analysis (GSEA) A method to test if predefined sets of related genes (e.g., common function, regulation) are enriched, that is, statistically over-represented in a given set of genes, most often from a differential expression experiment.

genomics Biology and methods focused on an organism’s DNA (i.e., its genome).

genotype The complete genomic identity of an organism from an information point of view.

graph database A database designed to store and query data structured as nodes, edges, and properties, allowing efficient representation and traversal of complex relationships.

hallucination A phenomenon whereby large language models generate outputs that are factually incorrect, fabricated, or not grounded in the provided data or context.

human-in-the-loop A model of interaction where human feedback is integrated into the AI system’s decision-making process, ensuring that AI outputs align with human expertise and ethical standards.

in-context learning Also, *prompt-learning*, is a form of few-shot learning where an LLM prompt includes a few examples (the context) followed by a question. The context effectively primes the LLM to make an inference when answering the question. Contrast with zero-shot learning.

knowledge graphs Data structures that represent information as a network of entities (nodes) and their relationships (edges), allowing AI systems to reason about and retrieve information based on these interconnected concepts.

large-language model A type of artificial intelligence model that processes and generates human-like text based on vast amounts of language data, using techniques such as deep learning and transformer architecture to understand and produce coherent language outputs.

latent Dirichlet allocation A method for automatically discovering (and assigning) topics across a group of documents.

machine learning A class of computer algorithms typically trained on a data set such that they learn to make predictions on previously unseen similar data. May be supervised (i.e., training data classes are labeled), unsupervised (i.e., structure is found in unlabeled data), or some combination of the two.

mRNA Messenger RNA is the final product of transcription, the mature transcript, which is then ready to be translated (i.e., synthesized) into a protein.

multi-modal Systems or models that can process and integrate multiple types of data, such as text, images, and audio, enabling more comprehensive understanding and interaction with complex information.

NoSQL database A type of database designed for storing and retrieving unstructured or semi-structured data, offering flexibility in data models and scalability, often used in large-scale data-driven applications.

pathway enrichment analysis A bioinformatics technique used to determine if a set of genes, often derived from differential expression studies, is significantly over-represented in known biological pathways, providing insights into the underlying biological mechanisms.

phenotype The physically observable identity of an organism as an expression of its genotype in some environment.

positional encoding A technique used in transformer models to encode the order of input tokens by adding position-specific information to each token, enabling the model to process sequences without inherent order.

proteomics Biology and methods focused on an organism's proteins (i.e., its proteome).

recurrent neural network A type of neural network designed to process sequential data by using connections that form directed cycles, enabling it to retain information from previous inputs as context for current processing.

retrieval-augmented generation A method that combines generative language models with information retrieval systems to improve the accuracy and relevance of generated responses by accessing external databases or documents during the generation process.

RNA Ribonucleic acid. A single stranded molecule originating as a transcript from DNA. This molecule undergoes various modifications (e.g., splicing) before being translated into protein. Composed of four bases: adenine (A), cytosine (C), guanine (G) and uridine (U).

RNA-seq A next-generation sequencing technique for collecting RNA from a biological sample of cells.

scRNA-seq A sequencing technique for collecting RNA from single isolated cells.

self-attention A mechanism within transformer models that allows each element of an input sequence to weigh and attend to all other elements, enhancing the model's ability to capture contextual relationships.

semantic analysis The process of understanding the meaning and relationships of words and phrases in text, allowing AI systems to comprehend context, resolve ambiguities, and generate more accurate responses.

serverless A computing environment where the infrastructure management, including server provisioning, scaling, and maintenance, is handled automatically by the cloud provider and dynamically allocating resources as needed, charging only for the actual compute time used during API requests.

supervised learning A machine learning approach where models are trained on *labeled* data, using input-output pairs to learn a mapping from inputs to desired outputs.

transcriptomics Biology and methods focused on an organism's RNA (i.e., its transcriptome).

transformer architecture A deep learning model architecture that uses self-attention mechanisms to process input data in parallel, enabling more efficient and accurate processing of sequences like text, and is the foundation of many advanced AI models, including large-language models.

unsupervised learning A machine learning approach where models are trained on *unlabeled* data to identify underlying patterns, structures, or groupings without predefined outputs.

vector database A specialized database optimized to store and retrieve high-dimensional vector embeddings, often used in applications like similarity search, recommendation systems, and natural language processing.

zero-shot learning In the context of LLMs zero-shot learning implies that a question is asked for which the model has not been explicitly trained. Thus, the model must generalize based only on its training data without any examples. Contrast with in-context learning.

LIST OF ABBREVIATIONS

adjP adjusted p-value

AI artificial intelligence

API application programming interface

BERT bidirectional encoder representations

CBR case-based reasoning

CCM critical care medicine

cDNA complementary DNA

CLI command line interface

CNN convolutional neural network

CoT chain-of-thought

CRAG clustered retrieved-augmented generation

CRC colorectal cancer

DE differential expression

DGE differential gene expression

DL deep learning

DNN deep neural network

ENCODE encyclopedia of DNA elements

GAN generative adversarial network

GEP gene expression profile

GPCR G protein-coupled receptor

GPT generative pre-trained transformer

GSEA gene set enrichment analysis

GUI graphical user interface

HUGO Human Genome Organization

JSON JavaScript object notation

KG knowledge graph

LDA latent Dirichlet allocation

LLM large language model

LmRaC language model research assistant & collaborator

logFC log fold-change

MEF mouse embryonic fibroblast

MeSH Medical Subject Headings

ML machine learning

mRNA messenger ribonucleic acid

NCI National Cancer Institute

PEA pathway enrichment analysis

PMC PubMed Central

RAG retrieval-augmented generation

REST representational state transfer

RNA ribonucleic acid

RNN recurrent neural network

T2DM type 2 diabetes mellitus

TCGA The Cancer Genome Atlas

TME tumor microenvironment

XAI explainable artificial intelligence

YAML YAML ain't markup language

XML extensible markup language

CHAPTER 1 INTRODUCTION

The development of Large Language Models (LLMs) has revolutionized artificial intelligence, providing powerful tools for text generation, natural language understanding, and multi-modal interactions. However, despite their remarkable capabilities, LLMs have primarily been utilized as standalone tools or misinterpreted as vast databases of knowledge, which has limited their effective integration into complex, real-world applications, particularly in scientific research and critical domains like healthcare. This misuse has exposed significant challenges, including issues with transparency, explainability, and accountability, which are critical in fields requiring high levels of reliability and ethical standards.

Following the recommendation of a recent commentary in *Nature Medicine* [12] this research aims to address these limitations by reimagining the role of LLMs within a hybrid system that treats them not merely as knowledge repositories but as sophisticated language reasoning engines. The following research integrates LLMs with a retrieval-augmented generation (RAG) architecture and makes it available within a user note-taking application facilitating a dynamic, goal-oriented, and collaborative approach to problem-solving. This integration enables the seamless incorporation of quantitative data and the development of a transparent and traceable reasoning process, ensuring that the LLM's outputs are both reliable and explainable.

The significance of this research lies in its innovative approach to embedding LLMs within a broader software system that enhances the accuracy and efficiency of scientific discovery and healthcare decision-making. By focusing on the development of a

collaborative framework that supports human reasoning processes, this work addresses the current shortcomings of LLMs, offering a more accountable and ethically sound application of AI. Additionally, the research leverages interdisciplinary methodologies, including cognitive science and psychological principles, to improve the interaction between human users and LLMs, thereby creating a more intuitive and supportive environment for complex problem-solving.

In summary, this work establishes a new path for the integration of AI in critical scientific and research applications by developing a hybrid system that leverages the strengths of LLMs while addressing their limitations in transparency, reliability, and ethical use. This approach has broad implications for the future of AI in science, healthcare, and beyond, and, hopefully, sets the stage for more effective, responsible and productive use of these powerful technologies.

1.1 Statement of the Problem

Despite the significant advancements in large language models (LLMs) for tasks such as text generation and natural language understanding, their integration into broader software systems, particularly for complex scientific and research applications, remains limited. Current LLM applications often treat LLMs as standalone tools or, wrongly, as vast databases of human knowledge. Additionally, LLMs are frequently used as black-box components, leading to issues with transparency, explainability, and accountability that render them unsuitable in critical domains such as healthcare and scientific research. There is a need for a hybrid system that leverages the strengths of LLMs for language reasoning while ensuring they are part of a transparent, collaborative, and human-centric

solution-based system that integrates quantitative data and supports reasoning processes in a goal-oriented workflow. This research addresses these gaps by designing an LLM-integrated note-taking framework that enhances the accuracy, efficiency, and ethical use of AI in scientific discovery and beyond.

1.2 Research Objectives

1.2.1 LLM as Computational Component

Objective: Design a hybrid systems leveraging the strengths of LLMs for language manipulation while deeply integrating with traditional software methodologies and architectures.

Approach: DO NOT use the LLM as a knowledge-base/database. DO use it as a sophisticated language reasoning engine integrated into a case-based reasoning (CBR) and RAG-based architecture utilizing a human-in-the-loop philosophy.

1.2.2 Explainability and Reliability

Objective: Demonstrate that an LLM can be part of a system with a high degree of explainability (i.e., how answers are arrived at) and reliability (i.e., that performance is consistent and complete across a wide variety of problems).

Approach: Not only are LLMs opaque in how they arrive at answers, but they can provide convincing and plausible answers from fictitious authoritative references. RAG techniques will ensure that information is authoritative, current and complete and that the reasoning process itself is transparent and traceable. CBR will ensure that solutions are traceable and adaptable to novel problems.

1.2.3 Dynamic Information & Reasoning

Objective: A data environment that is dynamic and evolving and part of a solution oriented system.

Approach: Current LLMs are trained at a point in time. The scientific process is dynamic and evolving. Smoothly integrate quantitative data with language based tasks. New information, especially from experiments, will be easily incorporated and accessed while old information can be revised, reinterpreted, replaced. Reasoning is not static recall, but a goal oriented multi-step process, itself revisable. CBR is a well established approach to achieve this objective.

1.2.4 Collaborative Note-taking

Objective: Create an environment where a user can manipulate, transform and interact with both primary and derived information and data to create a seamless collaborative experience.

Approach: The office "whiteboard" as metaphor for collaboration, brainstorming and problem solving. Integrate the language-data LLM reasoning system with a human note-taking system in much the same way that a whiteboard is used to dynamically capture ideas, facilitate productive interaction, and flexibly adapt to new and evolving ideas. Emphasize the collaboration between human and machine as partners in reasoning and finding solutions to problems. Expose all major components of the system (notes, tools) as first-class notes available to user and system alike.

1.3 Organization

Chapter 1 (this chapter) introduces the problem and objectives of this research.

Chapter 2 provides a review of background topics relevant to this work as well as related research.

Chapter 3 gives the overall design and methodology of the LmRaC project as well as the progression of releases and their functionality.

Chapter 4 goes into detail on the first release and its foundational design: LmRaC Command Line [13].

Chapter 5 details the next release: LmRaC Web Application. This chapter also provides detailed usage documentation for the downloadable LmRaC Docker container [14].

Chapter 6 covers recent early integration of LmRaC with the Obsidian note-taking application: LmRaC Collaborative Notes [15].

Chapter 7 covers recent prototyping work demonstrating the LmRaC Collaborative Notes application as a foundation for scientific workflow for real-world scientific discovery [16].

Chapter 8 provides a unified discussion of the three major LmRaC research stages. This includes details on: overall results, usability, performance, limitations, technical challenges and its relationship to other current research.

Chapter 9 discusses innovations and the significance of the research. It also examines both the design philosophy as well implications for ethical computing.

Chapter 10 provides a roadmap for future work building upon the LmRaC research project.

1.4 Disclaimer

As of this writing Chapters 4 and 5 report on completed and published work for which software is available for public download. Chapters 6 and 7 report on recent research work that is currently undergoing peer-review. The author makes no representation that this recent work is complete as much of it is still in a rough proof-of-concept phase.

CHAPTER 2 BACKGROUND AND RELATED WORK

2.1 Large Language Models (LLM)

Today's large language models have deep roots in the earliest days of back-propagation learning which laid the foundation for Recurrent Neural Networks (RNNs) [17] and Long Short-Term Memory (LSTM) networks for sequence modeling [18]. Self-attention mechanisms were then introduced (2017) that improved models by allowing them to focus on different parts of an input sequence [1]. This architecture based on attention, called a transformer, allowed for attention to be parallelized setting the stage for more advanced models like BERT [19] – which leveraged bidirectional context for better language understanding – and generative pre-trained transformers (GPTs) like OpenAI's GPT-3 [20] and GPT-4 [21]. Both BERT and GPT-4 are instances of large language models based on the transformer architecture, however, BERT is designed more for language comprehension tasks while GPTs are designed to generate language.

The transformer architecture (Figure 1), though originally designed for parallelism (unlike RNNs), also introduced two other key features: positional encoding of words and self-attention. Positional encoding allows word order to be explicitly encoded with the word, rather than be implicit in the sequence of processing as they are in RNNs, thus allowing all words to be processed in parallel (Nx). Typically, position is encoded by adding a vector generated from sine and cosine functions of varying frequency. This effectively translates position into time. This vector is then added to an embedding vector for each word. Self-attention, on the other hand, establishes a relationship between all words in the input. Multiple self-attention heads, each with distinct weights, are used to capture

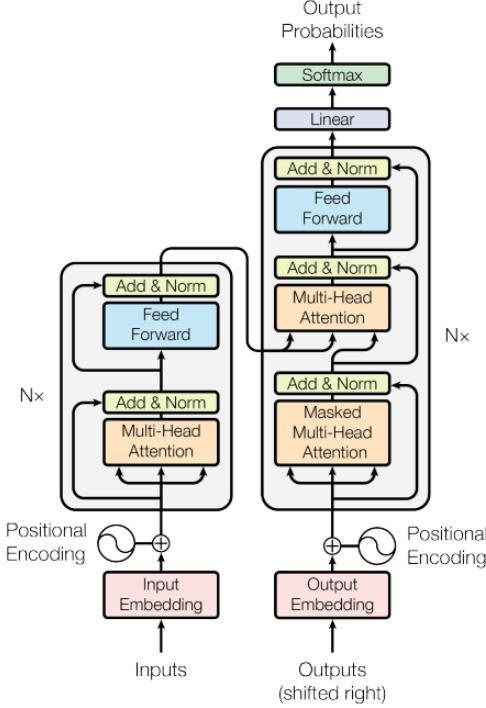


Figure 1: The Transformer model architecture (Figure 1 from [1]). The left side represents the encoder. The right side represents the decoder.

different relationship types. Attention vectors are then added to the word-position vectors using a feed forward technique to complete the encoding. Decoding operates similarly but with added masking to disallow lookahead to future tokens. To allow the decoder to focus on relevant parts of the input, cross-attention is added between encoding and decoding as output tokens are generated. Note that self-attention uses a *query-key-value* approach for shifting attention focus. The *query* represents the current state. Queries are compared to *keys* which yield the *value*. Values then represent the output of the attention mechanism. The final *Linear* and *Softmax* blocks of decoder map the output vectors into vocabulary space and then assigns a probability from which the best token (word) is selected.

Recent surveys by Zhao et al. [2] provides a detailed and extensive review of the current and rapidly evolving landscape of LLMs (Figure 2). Of particular note is the

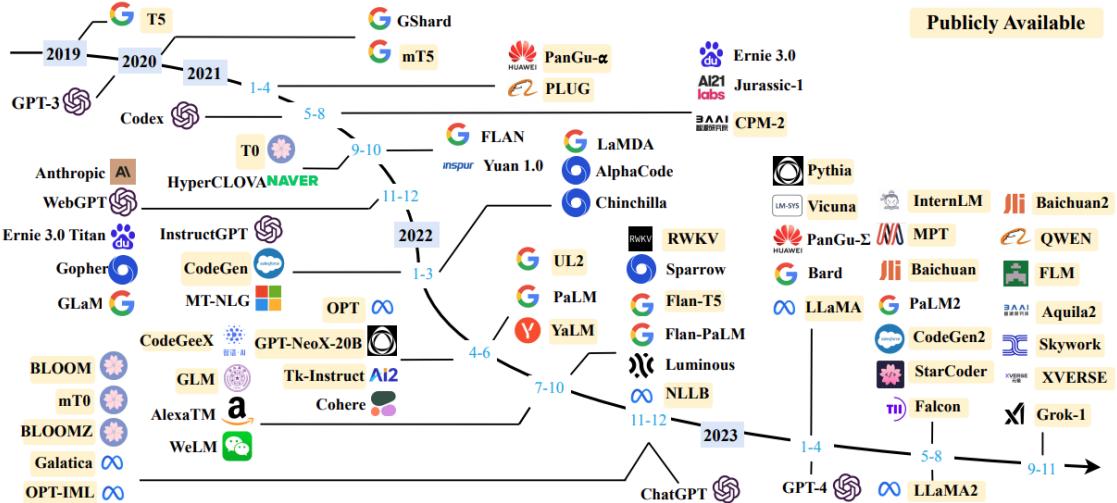


Figure 2: A timeline of LLMs larger than 10B parameters (Fig. 3 from Zhao et al. [2]).

rapid progression from: statistical LMs for specific tasks (1990), to neural LMs for language feature learning (2013), to pre-trained LMs for context-aware natural language tasks (2018), to, finally, LLMs for general-purpose task solvers (2020). LLMs were the first language models to display emergent abilities, that is, abilities not present in smaller models. These include: in-context learning (i.e., learning from a few examples), instruction following, and step-by-step reasoning. Current major issues with even the best LLMs include: under-performance of generation for specialized domains or tasks; hallucination, both intrinsic (information in conflict with existing sources) and extrinsic (unverifiable information); knowledge that is up-to-date; reasoning inconsistency, especially in step-by-step scenarios; generalization of numerical computation.

Minaee et al. [3] reviews three popular families of LLMs: OpenAI’s GPT, Meta’s Llama, and Google’s PaLM. It also provides an overview of how LLMs are built (Figure 3) and how they can be augmented. Benchmarks and popular LLM datasets are also reviewed.

With respect to usage, LLMs are *prompted*, rather than programmed. The prompt

provides, after pre-processing, input to the transformer's encoder. This results in an output *response*. Although responses from the decoder are tokens, depending on the model post-processing can render tokens in a more structured format, typically JSON. Prompt engineering assumes the role of the "programming" step enabling the user to craft the desired input-output characteristics. This can include constraints imposed upon the LLM as well as examples of input-output pairs (often structured as JSON). This allows the LLM to "learn" based on a few examples, called one-shot or few-shot learning. In the case where no examples are given and the user wants the LLM to generalize based solely on its training this is referred to as zero-shot learning.

Finally, LLMs have been applied to an extraordinarily diverse array of domains, including: customer support [22], personal assistants [23], healthcare [24, 25], education [26, 27], entertainment [28–30], legal [31], business/finance [32–34], marketing/advertising [35, 36], thematic analysis [37], creative arts [38, 39], mathematics [40], gaming [41] , and scientific literature review [42–47]. This last is particularly relevant to the work proposed herein.

2.2 Retrieval-augmented Generation (RAG)

Originally developed to enhance GPTs with the ability to retrieve information from external databases or documents [48], RAG architectures have evolved significantly to: increase the breadth of knowledge available, integrate real-time data sources, increase overall scalability, ensure the accuracy of retrieved information, and seamlessly integrate this information into the LLM's generative process. Models like REALM [49] have demonstrated the effectiveness of externalizing information by relying on a document

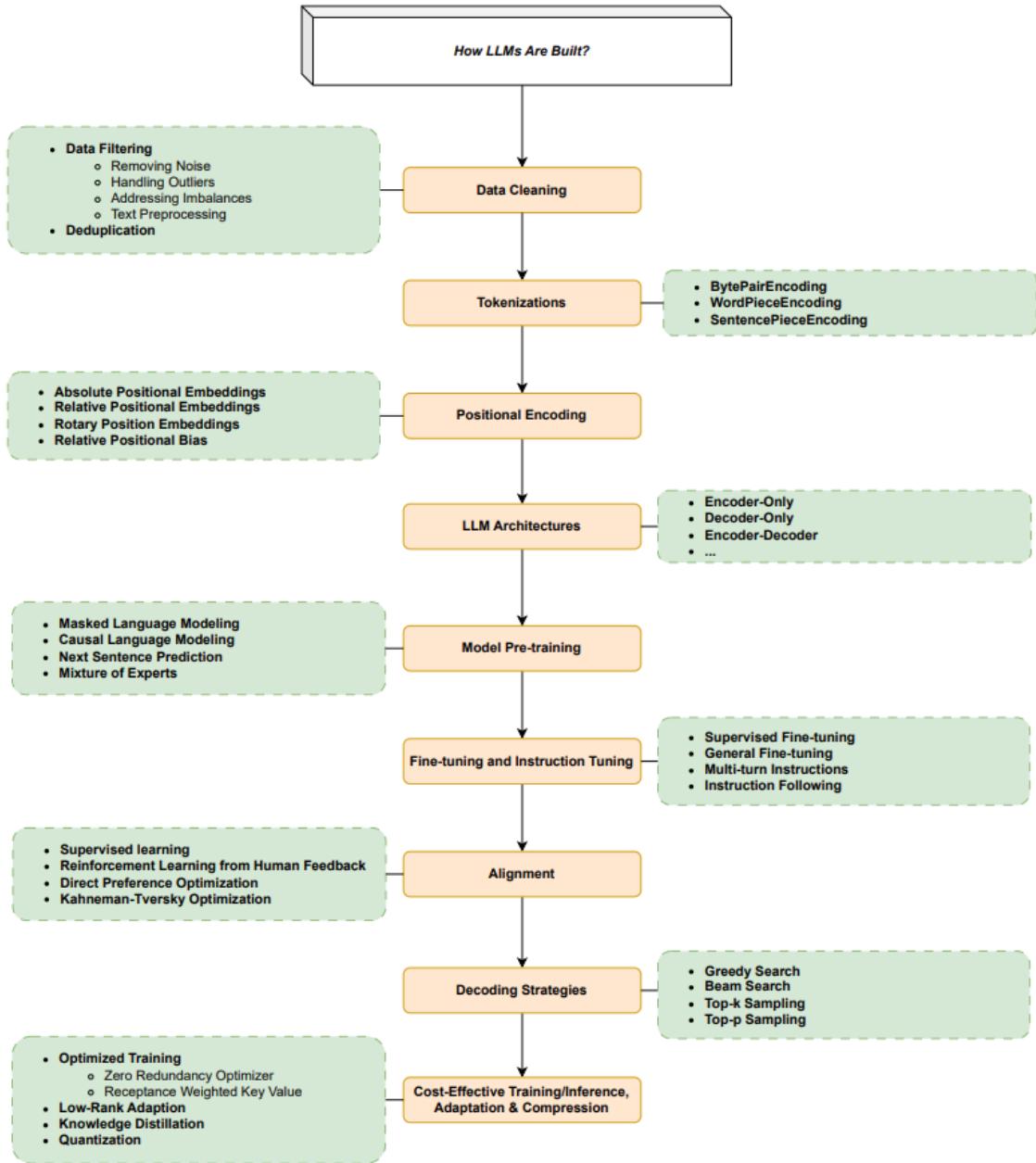


Figure 3: The different steps and components of an LLM when being built (Fig. 25 from Minaee et al. [3]).

retriever during LLM inference. This allows a large corpus of documents (e.g., Wikipedia) to be explicitly retrieved instead of implicitly stored in the model’s network weights.

Gao et al. provide a review of current RAG paradigms, including: naive RAG, advanced RAG and modular RAG [50]. Naive uses a indexing-retrieval-generation process to chunk sources, retrieve chunks based on some similarity measure, and presenting the selected results for generation of a response. Advanced RAG applies techniques to optimize retrieval concentrating on improved indexing (and use of metadata) and use of improved ranking of retrieved data. Modular RAG essentially adapts steps in the traditional RAG process to the task at hand (e.g., using search to refine retrieval). Also described are various retrieval strategies that iterate, recurse or adapt to improve the quality of retrieved knowledge.

Other variations and refinements of RAG designed to enhance the architecture’s efficiency, accuracy, and utility across various applications, include: Speculative RAG which uses a generalist LLM to verify multiple drafts generated by smaller specialist models as a more collaborative process [51]; combining RAG with Chain-of-Thought to improve reasoning for a logically consistent response [52]; Stochastic RAG which optimizes the retrieval process by maximizing stochastic expected utility (i.e., highest average performance over time) [53]; various chunking optimization techniques [54–56]; augmentation with metadata to improve the relevance and accuracy of retrieval by using contextual information; Clustered Retrieved Augmented Generation (CRAG) which leverages clustering, summarization, and aggregation techniques to reduce the number of tokens used for retrieval while maintaining high accuracy [57]; and, PolyRAG which introduces a pyramidal multi-layered querying system that integrates

ontologies, knowledge graphs (KGs), and chunk-based raw text to allow for more nuanced and comprehensive retrieval [58]. Finally, the development of prototyping and experimentation workbenches for RAG, like RAG Foundry, provides researchers and developers with tools to experiment with and refine RAG models [59].

2.3 Agents, Reasoning and Planning

Although LLMs have proven adept at quickly, if not always accurately, answering questions from an encyclopedic array of knowledge, they have been shown to have only a limited ability to provide answers that require multiple reasoning steps and planning. The ability to explore multiple lines of thought that may or may not be fruitful or may even be self-contradictory has been an area of intense research recently.

One foundational approach to solving complex tasks with LLMs is agential architecture which has its roots in non-LLM-based artificial intelligence [60, 61]. In this context "agent" broadly means a distinct actor designed to address one component of a larger problem [62]. Each agent may be assigned a specific role or may be imbued with specialized capabilities. Generally, an agential approach is cooperative and coordinated [63–65], however, competition may be employed as well to create a negotiated result. This can be especially important when evaluating potential adversarial attacks on a distributed system [66]. Combining agential approaches to include reasoning with acting [67] and add active web search [68] to supplement LLM knowledge is an active area of research designed to enable multi-step reasoning.

Since the early days of transformers, integral to reasoning, planning and problem solving have been data structures (knowledge graphs or KGs) that both store and structure

knowledge in a way that enhance LLM efficiency and functionality [69, 70]. These semantic graphs typically represents knowledge using a node-edge structure to describe the relationship (edges) between concepts (nodes). Both nodes and edges can also have associated metadata to "type" both the concept and relationship (e.g., "person" node; "is-a" relationship). The next step beyond using knowledge graphs has been to leverage LLMs to dynamically construct them as part of the reasoning task [71]. KGs can also be integrated into a hybrid architecture with RAG such that the KG is used to improve RAG navigation and retrieval [72]. Also, as LLMs have become multimodal (i.e., include not only text, but images, sound etc.) so have KGs as a method of enhanced reasoning [73].

Hand-in-hand with agential architecture are methods and techniques for coordination and planning. Systems like REAPER highlight the planning aspect, generating retrieval plans in conversational systems for better performance and scalability [74]. Chain-of-Thought Prompting [75] has been an influential approach for enhanced reasoning and decision-making capabilities with LLMs . By breaking down complex tasks into more manageable intermediate steps, aka thoughts, performance on the original task can be greatly improved. And, not only is performance improved, but transparency of reasoning becomes integral to the process as it now becomes possible to understand how the LLM arrived at a particular response through this chain. Tree of Thoughts [76] generalizes this approach to allow for multiple different reasoning paths. By having multiple paths, it becomes possible to support multiple strategies, including: next best choice, look-ahead, backtracking and global best choice. Approaches like HuggingGPT [77] employ an LLM as "executive" to choose agents at the model level by leveraging the Hugging Face repository of LLM models [78]. Not only does this foster choosing the best model for the

job (multi-domain), but, implicitly, it enables multi-modal problem solving. Chameleon [79] composes a sequence of tools, using their description and examples of use, to best generate a response to a query. These were tested on two large math and science problem sets: ScienceQA [80] and Tabular Math Word Problems (TabMWP) [81]. Recent surveys have begun to provide a taxonomy of this rapidly evolving literature of reasoning [82].

Another key to advancing reasoned problem solving is the introduction of feedback. Specifically, to overcome issues of consistency and, hopefully, to converge on a "best" response, techniques are being developed to allow an LLM to self-evaluate its output (feedback) and then use this to re-answer the question (refine). This technique was first explored in SELF-REFINE [83] which found the self-feedback usually actionable in further improving output in multiple domains (e.g., code optimization, mathematical reasoning). Systems like Reflexion [84] persist the feedback in episodic form so that subsequent questions can take advantage of previous feedback. Both systems leverage the language capabilities of an LLM to effectively improve performance without re-training. Recent work generalizes this approach and reviews how these and other techniques can address the larger issue of internal consistency of reasoning in LLMs [85].

Finally, LLMs are reaching beyond language to incorporate what can broadly be called tool use. Similar to how RAG architectures make external knowledge available to the LLM, tools make external functions available. Systems like Gorilla [86] allow LLMs to select from APIs and use documentation to correctly formulate their use in performing some task. Not only can the system utilize documentation, but it can recognize changes and, therefore, updates to an API. MM-REACT [87] integrates an LLM with tools for vision processing. This allows the LLM to choose the appropriate tool for a task and, in the

process, making the LLM multi-modal. In addition, tools can be coordinated by multiple agents to effect a cooperative solution to a problem. More general systems for LLMs to choose from a vast number of APIs are currently being investigated [88–92].

2.4 Scientific Discovery and Biology

The modes and methods of human problem solving and scientific discovery have a rich history of study [93–97] including those at the intersection of human and artificial intelligence [98]. This has led to study and work inspired by cognitive science to compare LLM analogical reasoning with humans and to improve this characteristically human skill in LLMs [99–103].

Although there have been and continue to be instances of LLMs being trained specifically for use in scientific domains including BioGPT [104], BioBERT [105], SciBERT [106] and others [107, 108] and even genomic DNA as a "language" [109, 110], the use of RAG architectures for scientific research and discovery have, until recently, often centered on improving question and answer performance (i.e., chat) [111–113]. Work like RUGGED [114] leverages LLMs to create a RAG-based workflow for hypothesis generation in the domain of drugs and disease.

In science BioPlanner is an innovative tool that generates experimental protocols using LLMs. [115]. PaperQA leverages a Retrieval-Augmented Generation (RAG) framework to retrieve full-text scientific articles and provide accurate answers to complex queries, demonstrating the utility of LLMs in enhancing research efficiency [111]. Recently there have been larger efforts to both facilitate scientific discovery as well as provide benchmark datasets to evaluate performance. DeepSpeed4Science is a large multi-institutional

initiative that employs LLMs to accelerate scientific discovery across disciplines [116]. LAB-Bench and SciKnowEval are benchmarks designed to evaluate LLMs' capabilities in biology and chemistry, respectively. LAB-Bench consists of 2,400 questions assessing tasks such as literature search and data analysis [117], while SciKnowEval offers 50,000 multi-level problems to evaluate LLM performance in scientific contexts [118].

Healthcare has seen an exponential increase in interest in the use of AI. LLMs are being utilized in critical care medicine (CCM) for decision support, medical documentation, and improving doctor-patient communication. These applications demonstrate the potential of LLMs to enhance patient care and streamline clinical workflows [119]. Furthermore, MedVersa is a multimodal system that integrates LLMs for learning and tasking in medical imaging, showcasing their versatility in handling diverse medical data [120]. In oncology, LLMs have been employed to match patients to clinical trials and support precision oncology by interpreting genetic data and planning personalized treatments [121, 122]. CancerGPT uses few-shot learning to predict the synergy of drug pairs for treating cancers in rare tissues, illustrating the advanced capabilities of LLMs in oncology research [123]. Significant advances have also been demonstrated in leveraging the foundational language abilities of LLM in electronic health records (EHR). A scoping review of LLMs processing EHRs highlights their effectiveness in extracting and synthesizing critical information, thereby supporting clinical decision-making [124]. LLMs have also outperformed experts in summarizing clinical texts such as radiology reports and patient progress notes [125]. A case study demonstrated the efficacy of an LLM-RAG pipeline in preoperative medicine, using guidelines to assist in clinical decision-making and showcasing robustness comparable to human responses [126]. Additionally, LLMs

have shown promise in improving differential diagnosis accuracy [127]. Evaluations of LLMs in clinical workflows, as opposed to answering standardized test questions, highlight their practical utility in real-world settings [128]. MultiMedQA provides a benchmark for assessing clinical knowledge, moving beyond simple Q&A to practical applications [129]. DeepDR-LLM combines LLMs with image-based deep learning to offer individualized diabetes management recommendations [130]. LLMs also assist clinicians in responding to patient portal messages, enhancing patient-clinician interactions [131].

Several comprehensive reviews and guides have been published to help clinicians and researchers navigate the integration of LLMs in healthcare. These include primers on LLM technology, practical guides for non-computational cancer researchers, and summaries of the state of the art in healthcare applications, transitioning from Q&A to decision-making [132–134].

Much has been written about how best to use LLMs in the scientific domain, whether as sources of knowledge and reasoning or as tools and components integrated into current workflows. Concerns range from reliability and accuracy to safety and ethics. Driving much of the conversation is the well documented phenomenon of hallucination as well as the opacity of function due to both fundamental characteristics of LLMs as well as proprietary considerations. Interestingly, recent work has demonstrated the use of semantic entropy as a measure of uncertainty and therefore a possible indicator of confabulation when a set of answers show semantic inconsistency [135].

LLM applications like ChatGPT offer transformative opportunities in science by enabling more intuitive data interaction. Technology features in journals like *Nature* hold out the promise of "talking to your data" [136] while other editorials promise the clinical

researcher accelerated scientific discovery, enhanced collaboration, and improved data management practices [137].

Despite the potential benefits, several critiques and concerns about LLMs have emerged. One major issue is the lack of transparency in their underlying engineering, which frustrates researchers and raises questions about the reliability and reproducibility of LLM-driven research [138]. Specifically, in fields like chemistry and drug discovery, there are significant concerns about the autonomy of LLMs and the potential for misuse. Tang et al. highlight the risks associated with autonomous agents in science, particularly emphasizing the need for safeguards in chemistry and drug synthesis [139]. Systematic reviews elaborate on these ethical challenges, discussing issues such as bias, accountability, and the implications of LLMs on patient care and medical research [140] while other work underscores the importance of developing robust ethical frameworks [141].

LLM-based agents, such as ChemCrow, have accelerated concerns about the misuse of LLMs. ChemCrow integrates multiple expert-designed tools to perform tasks across organic synthesis, drug discovery, and materials design, raising alarms about the potential for these technologies to be exploited [142]. The development of CLAIRify, which automates chemistry experiments using robots guided by LLMs, and Coscientist, an AI system driven by GPT-4 for autonomous experiment planning and execution, further illustrates the dual-use dilemma of LLMs in scientific research [143, 144]. To mitigate these risks, several solutions have been proposed. SciGuard, for instance, acts as middleware to control and mitigate risks associated with chemical and drug synthesis, ensuring safer and more ethical use of LLMs in scientific research [145]. Such frameworks are essential for balancing the promise of LLMs with the need for stringent safety and ethical standards.

CHAPTER 3 RESEARCH DESIGN & METHODOLOGY

3.1 Research Design

Research for this project has progressed through multiple versions of the LmRaC application (Figure 4). Early work concentrated on question and answer reliant on a user populated domain knowledge. This knowledge was instantiated using multiple RAGs deriving knowledge from PubMed articles. Questions could be open-loop as well as close-loop. In the closed-loop configuration domain knowledge was also made available from user data (i.e., experiments). Answers relied on a novel usefulness metric to improve RAG performance.

This work was then extended to include a web application interface and user configurable REST API allowing flexible access to not only experimental data, but also user provided documents. Answers now included iterative reasoning to allow for incremental solving of problems. Though linear, this improved performance over single retrieval answer.

Recent work has moved from a web application to true user-centric interface by tightly integrating with the Obsidian note-taking application. This configuration achieves the overall goal of being a true collaboration between human and LLM. Reasoning is now based on cases which themselves are essentially a distillation of previous experience. Early results demonstrate that by making reasoning, in the form of cases, and tools first-class notes in the system it is now possible for the user to participate as an agent in the LLM’s reasoning.

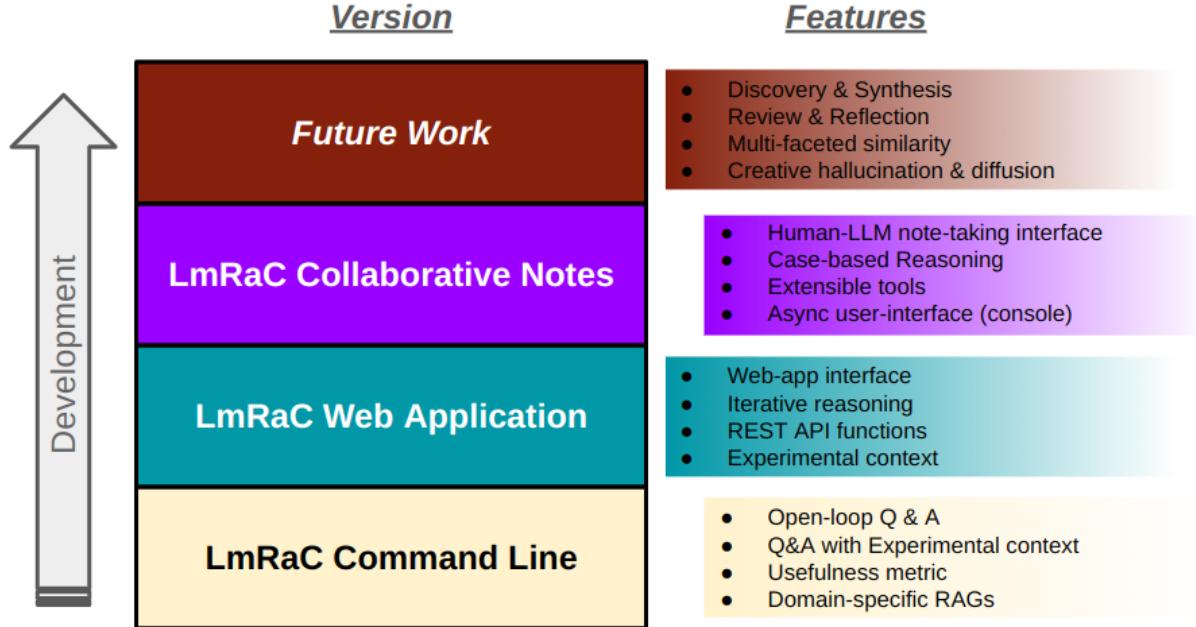


Figure 4: LmRaC research roadmap.

3.1.1 LmRaC Command Line

The initial version of LmRaC was a Python program where all interaction was through the console in command-line fashion. See Chapter 4 and "What's the Data Say? An LLM-based System for Interrogating Experimental Data," Craig & Draghici, 2024 [13]. This evolved from early prototyping in a Jupyter notebook to prove the RAG design. The command-line version also introduced LmRaC's tiered architecture for both open-loop and closed-loop question-and-answer. Three distinct knowledge domains (RAGs) were implemented: RAG_{dom} , to store the results of PubMed searches; RAG_{exp} , to store experiment specific documents and search results; and, RAG_{fun} , which retrieved data via a REST API to the user's data.

Open-loop Question & Answer

In the LmRaC command line version a user could ask a question and receive a reply. Unlike a chat, context is stateless and all answers are entirely derived using domain specific knowledge from the RAG_{dom} . Key to LmRaC's design philosophy the LLM is used only as a language-reasoning engine. Answers originate only from the RAGs enabling their source to be accurately cited down to the paragraph level.

Question & Answer with Experimental context

Experimental context was prototyped in the command-line version to demonstrate how domain knowledge culled from PubMed could be combined with user-specified knowledge originating from previous answers, or, ultimately, from user supplied documents.

Usefulness metric

An innovative feature of the command-line version of LmRaC was pre-screening of results from the RAG using a usefulness metric. This allowed LmRaC to answer questions using only source material that was most likely to yield an answer. The effectiveness of this strategy was demonstrated in the paper associated with this work.

Domain-specific RAGs

As mentioned above, command-line LmRaC laid the foundation for subsequent revisions by dividing knowledge into three distinct knowledge-bases: RAG_{dom} , RAG_{exp} and RAG_{fun} and demonstrated how this scheme could be used to augment authoritative sources with user experiment data.

3.1.2 LmRaC Web Application

The LmRaC web application moves from a single-thread command-line interaction to one that is more conducive to the back-and-forth expected between human and application. A main window allows the user to ask questions while sub-windows provide an intuitive interface to: indexes, functions, experiments and answer. See Chapter 5 and "LmRaC: A Functionally Extensible Tool for LLM Interrogation of User Experimental Results," Craig & Draghici, 2024 [14].

Web-application interface

Previous command-line functionality is wrapped in a Flask web server for the LmRaC web application so that interface components appear as web pages. This includes answers (which have always been output as markdown) which can now be browsed and rendered as either formatted HTML or as full-featured Markdown (through the use of a 3rd party cross-browser plug-in).

Iterative reasoning

Unlike the LmRaC command line version which uses a single question-answer cycle, the LmRaC web application preserves state while answering questions by taking into account previous questions or answers. This is a crude version of Chain-of-Thought (CoT) [75] that, by incorporating these into new prompts, creates continuity and a basic memory.

REST API functions

The LmRaC web application formalizes the interface for external user-defined functions by providing a separate standalone Python project. This code is available to the user to develop custom REST API services whether for data access or for interface to external

database APIs.

Experimental context

True experimental context is available in the LmRaC web application. The command-line version prototyped this, but the web-app separates experiment indexes in Pinecone storing embeddings for experimental data and documents as well as context forming answers. For example, an experiment that collects differential gene expression results for a tumor-normal pair would benefit by having documents that summarize previous findings from the literature. These documents can currently be created from questions to LmRaC. Once answered, the answer can be chunked and stored and then available for secondary questions about the experiment in the context of previous finds as summarized in this document.

3.1.3 LmRaC Collaborative Notes

While previous work in neural networks and LLMs have shown the advantage of shared memory between computational components [146–150] we feel that a collaborative system where the human user is a first-class agent in an LLM-enabled system has the potential to significantly advance reasoning systems and enhance scientific discovery beyond previous work [151–153]. For implementation and results see Chapter 6 and "Advancing Scientific Workflows: A Human-LLM Note-Taking System with Case-Based Reasoning," Craig, *forthcoming* 2025 [15]. For a real-world case study of this system and its application to scientific discovery see Chapter 7 and "A Human-LLM Note-Taking System with Case-Based Reasoning as Framework for Scientific Discovery," Craig, *forthcoming* 2025 [16].

Human-LLM note-taking interface

LmRaC's integration with Obsidian takes interactive documents one step further by having a shared-persistent memory architecture based on these documents (i.e., notes). LmRaC is no longer a standalone application, but an integral portion of human workflow. Not only are user notes shared but the main components of LmRaC are available to the user as first-class notes: case definitions and tool specifications.

Case-based reasoning

This component builds upon the LmRaC web-app by expanding on the notion of experimental context as a memory store of derived reasoning. Case-based reasoning notes act as both a store of knowledge (i.e., memory) and as a key interface of that store with the human agent. We feel this approach directly addresses many concerns with the use of LLM technology in healthcare and the clinical sciences (see [2.4](#)) by making the human integral to the overall system.

Extensible tools

Building on the extensible REST API interface of the LmRaC we application, tools (i.e., functions) are now wrapped in an Obsidian note. This makes their use and interface available equally to the user and system (i.e., LmRaC). Tools are now indexed as well. This means that rather than pre-load a group of functions to make them available to LLM prompts, functions/tools can now be found on-demand as the solution requires. This not only improves flexibility but also includes execution efficiency, since the case loads only those functions needed for any given prompt.

Asynchronous user-interface

Since LmRaC is no longer an application with its own interface, a "console" is provided to give the user a way to communicate when changing note text is not appropriate. This includes system functions (e.g., enable debugging), but also actions on notes (e.g., summary note, create outline from note). Because the console is event driven, it also facilitates interrupting during the application of case solutions. This flexibility allows the user to review, revise and even offer suggestions *while* a case is being executed, thus making the user an active agent in LLM reasoning.

3.1.4 LmRaC Future Work

Future LmRaC research includes: discovery and synthesis from notes; review and reflection of case execution and tool application; implementation of a multi-faceted similarity measure; and, creative hallucination and diffusion for CBR. Details of these components are discussed in Chapter [10](#).

3.2 Research Problem Domain

The biological sciences have been chosen as the problem domain in which to demonstrate the potential for designing an LLM-human collaborative application. This domain provides a rich set of problems, resources and datasets in which to develop tools useful for enhancing the scientific discovery process. The type of problems suitable for each phase of LmRaC has been chosen to exercise core feature for each successive software release. They are as follows:

LmRaC Command Line

A biological pathway is a series of interconnected biochemical reactions or molecular interactions where each step is driven by enzymes and other molecules to carry out specific functions, such as: metabolism, cell signaling, or gene regulation. Biological pathway analysis, particularly in the context of differential gene expression analysis, aims to identify and interpret the biological significance of changes in gene expression under different experimental conditions, such as disease versus a healthy control state. The purpose is to understand how these groups of genes are dysregulated in response to the condition. Methodologically, this involves first identifying differentially expressed genes (DEGs) through statistical analysis of expression data (e.g., from RNA-seq or microarrays), followed by mapping these DEGs to known biological pathways using databases like KEGG. The pathways that show significant change (i.e., enrichment) of DEGs are then analyzed to provide insights into the underlying biological processes, potentially revealing key mechanisms involved in the condition being studied.

The NIH National Library of Medicine's PubMed [154] provides a rich resource of peer reviewed papers for understanding both the underlying biology and associated experiments. We also utilize the commercial tool iPathway Guide [155] to generate DEG analysis results. Together these form a language and data rich domain in which to conduct our research of the command-line version of LmRaC.

LmRaC Web Application

The web application version of LmRac continues to explore biological pathways, but also more thoroughly supports experimental context. This context extends beyond user

data retrieved by the custom REST API to also include both documents selected by the user relevant to the experimental context *and*, closing the loop, results from previous LmRaC searches. This allows the user to build up a knowledge-base from not only primary PubMed sources, but derived and focused on some experimental purpose. Both experimental documents are indexed and made available through RAG search to aid a user in exploring a particular problem.

Question answering also makes explicit use of the iterative prompts to the LLM. This means that questions can be answered through an accumulation of response and data. This enables LmRaC to now supports asking questions that involve three levels of knowledge: general (from PubMed), data (from the REST API) and experimental (from the context of select documents and previous answers).

The interface also allows a user to more easily inspect answer, load data-specific functions, and manage knowledge sources through a more typical application interface.

LmRaC Collaborative Notes

The note-taking interface extends the collaboration metaphor further by moving LmRaC away from an application to be used, and toward features that truly extend the workflow of scientific discovery. This step of the research also moves question answering away from simple prompting to include memory of previous question-answer in the form of case-based reasoning. Leveraging CBR lays the foundation for the LLM to reuse and adapt previous solutions.

The interface of note-taking makes user interaction with LmRaC a natural extension of the thought process. It also facilitates interaction and collaboration by making the actions (cases) and data (notes) available equally to man and machine. Rather than throwing a

question over the wall and waiting for the system (LLM) to answer, the user can follow every step of problem solving, make suggestion, ask for explanation, request justification, revisit and revise steps in the process. This design is meant to move LLM use toward a more user-centric foundation where the strengths of LLMs go hand-in-hand with explainability, transparency, trust and ethical use of AI.

3.3 Implementation Details

LmRac research has evolved over the three releases using a variety of services, tools and languages. A brief overview of the major components is given here.

Prototyping for Preliminary Results

LmRac was initially prototyped in Jupyter Notebooks [156] running in Google CoLab [157]. Though not suitable for application distribution, Jupyter provides a self-documenting platform for rapid development and proof-of-concept testing. It also provides a good means to prototype components for later migration to full-stack development.

Development Environment

The web application of Obsidian [158] versions of LmRaC have been coded in the Visual Studio Code environment [159] using Python (v3.10.12) [160].

Command-line Executive

Preliminary design of LmRaC was implemented in Python 3. Python interfaces with all components described below through their respective API interfaces.

GPT-4

The LLM model for command-line executive function was *gpt-4-1106-preview*. OpenAI's API [21, 161]. Though other open source models (e.g., Llama) are available, OpenAI has

robust support for functional interface. That is, GPT4 has built-in support for structure interface (JSON) between the language model and external programming interface. Coupled with OpenAI's leadership in LLM performance, a design decision was made early on to use GPT-4.

GPT-4o

The web application and note-taking versions of LmRaC utilized the next release of GPT-4, 4o, released in May 2024. Although OpenAI has since released models o1, o1 pro and o3 mini, base functionality of these models with emphasis on autonomy, conflicts with LmRaC's design philosophy of human-in-the-loop for problem solving.

Vector Database

Vector embeddings are calculated using the OpenAI *text-embedding-ada-002* model with cosine distance. Vectors for paragraph embeddings are stored in a Pinecone serverless index [162]. Again, though open source solutions are available, Pinecone offers a serverless option that scales easily at minimal cost.

NoSQL Database

Full-text paragraphs from PubMed were initially stored in a Google Firestore NoSQL cloud database [163]. Firestore is a managed, scalable serverless database. With improved performance it was found that on-demand access of full-text did not significantly degrade LmRaC performance. The web application version began simply indexing paragraphs and then retrieving full-text from PubMed when needed. An MD5 check was added to detect any change to full-text since indexing (in which case the text is re-indexed).

Web Server

The LmRac web application was prototyped using the Flask framework [164]. Though not meant for production, it provides a fast prototyping environment suitable for distribution within a Docker container.

LmRaC Distribution

Distribution of the LmRaC web application executable was through Docker [165] and hosted on DockerHub.

Graph Database

neo4j [166] was used as the graph database for prototyping a simple semantic graph for use in conjunction with CBR.

Speech

Speech-to-text in the LmRaC console for note-taking collaboration uses the OpenAI whisper model [167]. For text-to-speech output from this LmRaC release OpenAI's tts-1 model was used [168]. WAV files are played using the Python Pygame library [169].

Scientific Journals

PubMed documents are accessed through the OA Web Service API [170]. PubMed document XML is parsed using Beautiful Soup [171]. Currently only publicly available full-text articles are used. A headless browser has been experimented with to allow full-text download of non-free publications which are available through personal or institutional subscriptions but is not currently integrated into LmRaC.

Obsidian and Markdown Notes

Obsidian [158] is a flexible note taking application that dynamically displays markdown documents. Initially, this had been used to view log files from LmRaC, but its full potential as an interface between human and LLM-based application is why it was chosen. The third phase of LmRaC has taken full advantage of Obsidian and is tightly integrated with note-taking workflow.

Performance Evaluation, Statistics and Visualization

To date all performance and statistical evaluation has been done in the R language [172]. Figure generation has been done in R using the ggplot2 visualization package [173]. No figures have been generated using AI other than incidental component icons (e.g., toolbox icon) using the ChatGPT DALL-E 3 [174].

Utility Programming

Preprocessing, data normalization and general big data wrangling have been performed using Perl [175].

CHAPTER 4 LMRAC COMMAND LINE

4.1 Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities across a variety of domains. However, issues such as hallucinations, implicit extrapolations, and inappropriate interpretations currently limit their wide adoption in scientific research. Retrieval-augmented Generation (RAG) has emerged as a promising technique to enhance the accuracy and reliability of LLM outputs. RAG integrates external information with LLM-generated content, ensuring more dependable results and enabling the inclusion of proprietary or domain-specific knowledge unavailable during the initial LLM training. In this work, we present an extensible LLM RAG architecture designed to answer complex scientific questions that can reference user-provided experimental data. The system uses domain-specific knowledge (RAG_{dom}) to respond to general inquiries, supporting answers with citations from authoritative sources. These responses can be recursively integrated to provide user directed context for experiments (RAG_{exp}), allowing subsequent questions to utilize both general and experiment-specific information. Our system can be easily extended with user-defined domain-specific modules for functionally interrogating quantitative experimental results (RAG_{fun}). We demonstrate its application in analyzing, summarizing, interpreting, and assisting with synthesizing results from a gene expression comparison of two phenotypes. The RAG_{dom} component encompasses biological pathways, genes, and disease-related knowledge. The RAG_{exp} component is derived from user queries of differentially expressed gene (DEG) and pathway enrichment analysis (PEA) results. Additionally, RAG_{fun} modules are designed to interface with DEG and

PEA data, enabling quantitative experimental results to be integrated seamlessly into user questions. Unlike traditional LLM outputs, our system’s results are free of hallucinations and include paragraph-level citations to the supporting literature. We compared the quality of our system’s output with those generated by LLM-only models such as GPT-3, GPT-4, Llama2, PaLM2, and BioGPT, and found our approach to be superior.

4.2 Introduction

Top scientific journals like Nature, along with every other corner of society, have taken note of the marvels of large-language models (LLMs) like GPT-4 [21] to not only carry on complex conversations (ChatGPT), but to perform scientific tasks with human-like performance. Editorials about LLMs transforming healthcare [176] and those expressing ethical concerns around transparency and the generation of seemingly authoritative false information [177] are published weekly. A common concern is the well documented phenomena of hallucinations [178, 179]. Another is the tendency of users in all domains to treat LLMs as omniscient repositories of knowledge. This is especially true in scientific domains [180]. Though trained on billions of pieces of information, LLMs are not designed to cite sources at even a minimal standard of academic rigor. Couple this with outright fabrication of facts and/or sources [181], and it is easy to understand the skepticism most scientists have with utilizing LLMs in any way that could undermine not only their work, but their career.

One response to the conundrum of what to do with an all-knowing yet unreliable partner has been the suggestion to use LLMs only as scientific reasoning engines [12]. This translates to leveraging their language reasoning skills while carefully constraining them

as a knowledge source. Instead, systems are designed to integrate knowledge *external* to the LLM. This can be done using techniques such as retrieval-augmented generation (RAG) while focusing on the strengths of the LLM to compare, summarize, extrapolate and evaluate from this external information as a reasoning engine. From a developer’s perspective, it also means designing LLM prompts that provide clear context for that reasoning and guardrails to prevent the LLM from injecting unreliable information.

We take that approach here demonstrating the LmRaC¹ (Language Model Research Assistant & Collaborator) architecture which utilizes an extensible multi-tier RAG design to allow users to interrogate their own experimental data in the context of an external domain-specific scientific knowledge base. We outline unique features of the architecture and demonstrate its application to systematically understanding how gene function is affected in real-world colorectal cancer data.

4.3 Related Work

Today’s large language models have deep roots in the earliest days of back-propagation learning which laid the foundation for Recurrent Neural Networks (RNNs) [17] and Long Short-Term Memory (LSTM) networks for sequence modeling [18]. Attention mechanisms were then introduced that improved models by allowing them to focus on different parts of an input sequence [1]. The Transformer model [1] then allowed for attention to be parallelized setting the stage for more advanced models like BERT [19] – which leveraged bidirectional context for better language understanding – and generative pre-trained transformers (GPTs) like OpenAI’s GPT-3 [20] and GPT-4 [21]. Both BERT and GPT-4 are instances of large language models based on the transformer architecture, however, BERT

¹Pronounced: Limerick, as in the verse form and Irish county.

is designed more for language comprehension tasks while GPTs are designed to generate language.

Originally developed to enhance GPTs with the ability to retrieve information from external databases or documents [48], RAG architectures have evolved significantly to: increase the breadth of knowledge available, integrate real-time data sources, increase overall scalability, ensure the accuracy of retrieved information, and seamlessly integrate this information into the LLM’s generative process. Models like REALM [49] have demonstrated the effectiveness of externalizing information by relying on a document retriever during LLM inference. This allows a large corpus of documents (e.g., Wikipedia) to be explicitly retrieved instead of implicitly stored in the model’s network weights.

Although there have been several instances of LLMs being trained specifically for use in scientific domains [104–107] the use of RAG architectures for scientific research and discovery is still relatively novel. Most uses have centered on improving question and answer performance (i.e., chat) [111–113].

Here we are interested in the integration of domain specific knowledge in the service of interrogating the results of a user’s scientific experiments. Specifically, we set the design goal of an easily populated domain knowledge base which can be extended on-demand (e.g., by searching the literature through a source like PubMed [154] or Google Scholar [182]). In addition, answers to questions about experimental results and data can be recursively fed back into the system.

4.4 Methodology

LmRaC can be thought of as a tiered architecture depending on the task at hand (Figure 5). At the lowest tier is the base architecture (Figure 5A) which functions like a typical question-answer system. The second tier (Figure 5B) adds feedback such that context specific answers become part of the knowledge base for further questions. Finally, a feed-forward loop (Figure 5C) adds functionality to retrieve experiment specific data and results as part of the question-answer cycle.

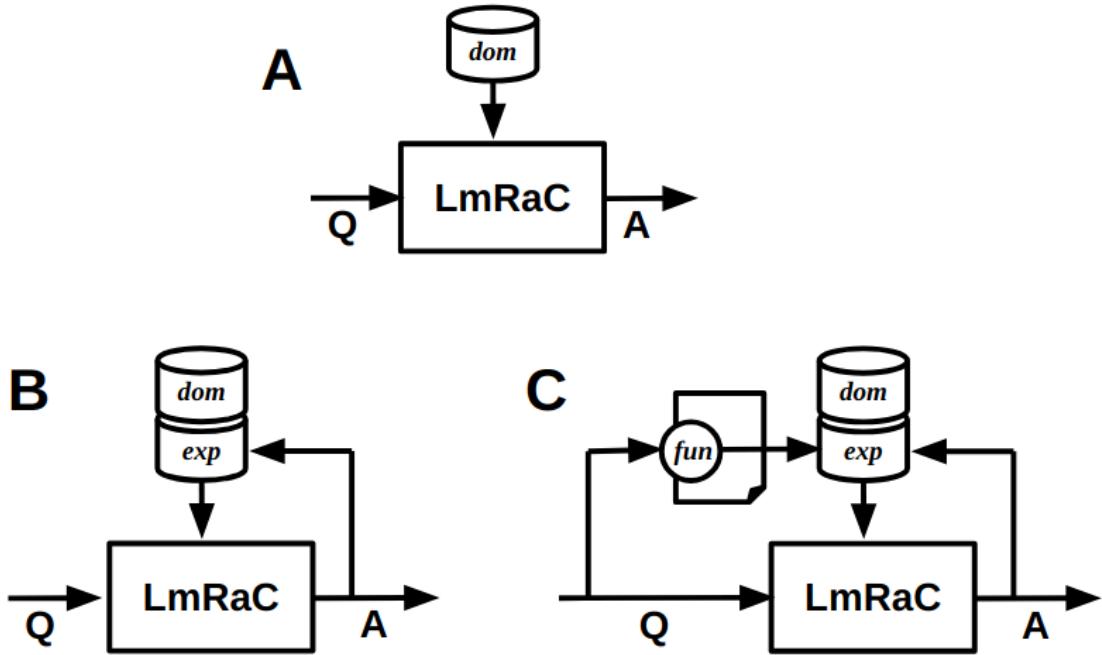


Figure 5: LmRaC architectures. (A) open-loop question and answer using domain specific information only (RAG_{dom}). (B) close-loop question and answer with experiment specific information (RAG_{exp}). (C) functional interrogation of experimental results (RAG_{fun}).

4.4.1 Domain Specific Question and Answer

The high-level flow of the LmRaC system is for a user to ask a question and receive a reply (Figure 6). Context is stateless (unlike a chat application). However, all answers are entirely based on a domain specific knowledge base. That is, the LLM

is used only as a language-reasoning engine generating answers solely on information retrieved from the domain knowledge. This retrieval-augmented generation of the domain knowledge (RAG_{dom}) ensures answers contain only cited sources of information from a user constructed authoritative domain.

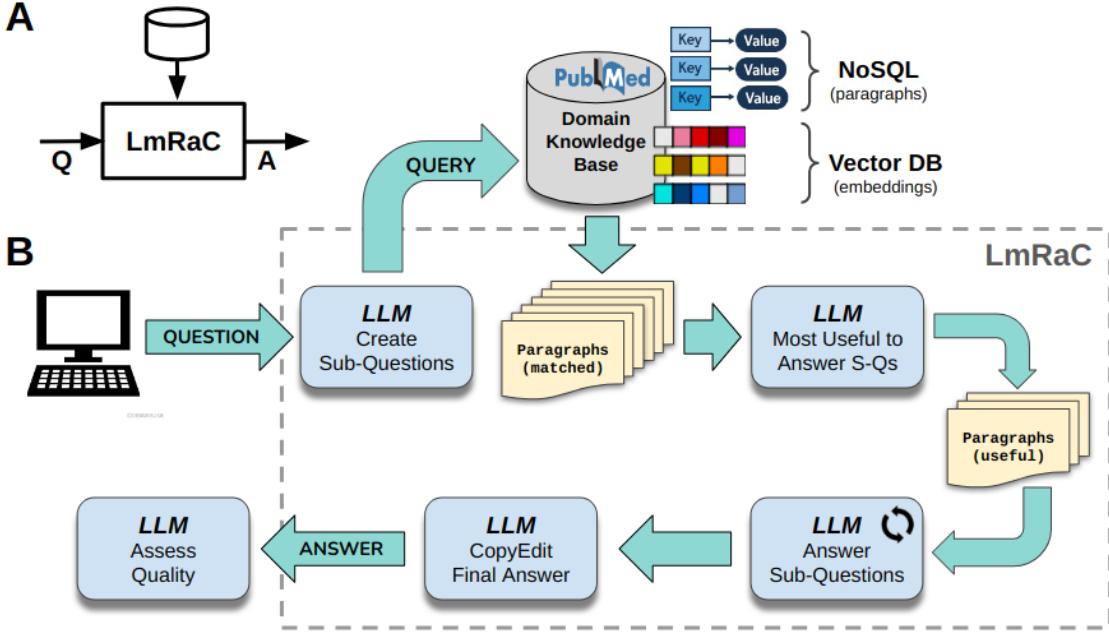


Figure 6: LmRaC base architecture. (A) high level open-loop question and answer, (B) detailed component implementation.

Domain Knowledge

Domain specific knowledge originates from user queries. That is, when the user asks a question LmRaC searches for domain specific terms (e.g., gene names, pathways, diseases). These are matched against previously loaded information using a controlled vocabulary (see Appendix A for vocabulary sources). If matching information is found the user is given the option to "update" it (e.g., re-run a PubMed search) or use what has already been saved. Journal articles, when downloaded, are parsed (XML) and split into paragraphs. An embedding is calculated for each paragraph and stored in a vector database. Any

associated metadata (e.g., from the search) is saved with the embedding vector. The full-text of the paragraph is saved to a NoSQL database as a key-value pair where the key is the associated document identifier and the value is the paragraph text. This allows the text to be quickly retrieved for any matched vectors. Collectively these two databases and their associated functions form the RAG_{dom} .

Sub-Question Generation

To ensure the user question is answered thoroughly and implicit questions are adequately explored, the LLM first generates a list of sub-questions derived from the original question and explicitly suitable for query of the vector database. The prompt is, in part:

PROMPT: *What series of queries to this database would best answer the question provided?*

The number of sub-questions also offers a mechanism for the user to control the thoroughness and complexity of the answer.

Vector Database Query

Each sub-question is converted into a vector embedding. Two separate queries are then performed: domain-wide and metadata filtered. The top k paragraph identifiers are then used to retrieve corresponding full-text from the NoSQL database. The mix of domain-wide and metadata filtered is designed to allow for knowledge discovery from unlabeled sources. Because all information is to some degree domain related, it is expected that information useful to any given question *may* come from a paragraph in an otherwise "unrelated" article (i.e., no metadata). The resulting set of paragraphs (i.e., chunks) form

the candidate set for answering any given sub-question.

Paragraph Usefulness Assessment

Candidate paragraphs are then assessed for their predicted usefulness in answering the sub-question. The prompt to the LLM is, in part:

PROMPT: *You answer user questions based on the research citations provided by the user above each question. Citation numbers are given in brackets. Grade how well each citation will contribute to answering the User Question. Grade each citation on a scale of 1 to 10. Do not answer the question.*

Paragraphs are then sorted base on their grade and only those receiving a 7 or higher are made available for the next step. If there are no useful paragraphs, then the sub-question is marked as "I don't know."

Sub-question Answer

Given a final set of "high quality" paragraphs found to be both semantically similar to the question and useful in the judgment of the LLM, LmRaC now prompts the LLM to answer the sub-question explicitly using only the supplied paragraphs (i.e., citations) and to properly cite the information that it uses.

PROMPT: *Based on the research citations provided by the user above each question. Provide bracketed numeric citations for any answer you provide. If the information can not be found in the research citations provided by the user you truthfully say "I don't know".*

Note that even at this stage it may not be possible to answer the question with the given

information. In that case a second attempt will be made. Failing that, the sub-question will be marked as "I don't know."

Answer Quality Assessment

Next, each sub-answer is assessed for how well it answered the question. This allows the LLM another chance to check its work and provide a quality answer.

PROMPT: *For the question given, assess the quality of the answer with respect to these citations, then provide a short explanation for your assessment.*

Only answers of "high quality" are then made available for final answer preparation.

Final Answer Preparation

All sub-question answers of adequate quality are concatenated and copy-edited while retaining citations using the following prompt.

PROMPT: *You are an expert copy editor. Edit the following text to remove repetition and improve readability and flow. Preserve bracketed citation numbers in your edit. Do not add any information and only remove information if it is redundant.*

Numbered citations in the final text are then replaced with a hyperlink to their underlying journal article. A final bibliography of references is prepared. A distinct advantage of chunking cited material by paragraph is that hyperlinks can then be qualified² at the paragraph level (see Appendix D for a full example).

²This utilizes the URL Scroll to Text Fragment qualifier (#:~:text=) introduced in Google Chrome and supported now in most browsers.

4.4.2 Question and Answer with Experimental Context

Feedback of answers, at the user’s discretion, means that the user specifies a context (e.g., experiment identifier) with which to tag the final answer. From this point the answer is processed similarly to journal articles: it is split into paragraphs, each paragraph has an embedding calculated, and then the full-text is stored in the NoSQL database. To allow for more fine-grained control of processing the embeddings are stored in a separate vector database. This ensures that derivative information is not mistaken for a primary source (recall that a domain-wide search is made for information discovery purposes).

The combination of this separate vector database and the associated collection of paragraphs constitutes the RAG_{exp} . When asking questions the user may ask general or context (i.e., experiment) specific questions. The choice of which knowledge source(s) to use – RAG_{dom} and/or RAG_{exp} – may be made explicitly by the user or inferred by the LLM from the question in much the same way that the RAG_{fun} is invoked. For more complex questions this means that LmRaC can effectively follow a simple chain of reasoning by answering a general question using the RAG_{dom} to gather background information needed to answer an experimental question (see *RAG Primacy Evaluation*, [4.6.4](#), below for how factual conflicts are handled).

4.4.3 Functional Interrogation of Experimental Results

Experimental results represent a special case for the domain knowledge base. Generally, the information is not in the form of documents, but, rather, in structured output from analysis or measurement devices or programs. Retrieval of this information depends on custom functions (see Appendix E for an example). One or more function

interface descriptions is provided to the LLM so that if, in its judgment, it needs additional information, it can return an intermediate request to LmRaC. LmRaC then calls the requested function and re-prompts the LLM with the original question along with the function's return results. The process can repeat any number of times for one or more functions until the LLM has retrieved the information it needs to answer the question. The set of functions and their interfaces to the LLM constitute the RAG_{fun} .

The RAG_{fun} employs a REST API to service any user-defined functions³. This makes the architecture fully extensible since a user need only provide LmRaC with an interface definition at *runtime*. For example:

```
FUNCTION getGeneExpression

PARAMETER geneArray:ARRAY
ITEM gene:STRING

PARAMETER filename:STRING*
PARAMETER experiment:STRING
```

This interface, along with a description of the function's usage (e.g., "Return gene expression results for a list of genes from an experiment") is sufficient for LmRaC to then make requests to answer questions related to gene expression.

4.4.4 Software Implementation

LmRaC is implemented in Python 3 using OpenAI's API [161]. The LLM model for executive function is *gpt-4o*. Vector embeddings are calculated using the *text-embedding-ada-002* model with cosine distance. Vectors for paragraph embeddings are stored

³A template API server is provided as part of LmRaC. User-defined functions are compiled into the server making them available to LmRaC via requests to the API.

in a Pinecone serverless index [162]. Full-text paragraphs are stored in a Google Firestore NoSQL cloud database[163] or retrieved on-demand from PubMed. Databases are accessed in Python through their respective APIs. PubMed documents are accessed through the OA Web Service API [170]. PubMed document XML is parsed using BeautifulSoup Soup [171].

4.5 Experiments

To demonstrate LmRaC’s performance we used results [183] from a larger comprehensive TCGA colorectal cancer study [184]. Differential gene expression results were loaded into iPathwayGuide[185, 186] to visualize these results as well as perform pathway enrichment analysis (PEA). The results are shown in Figure 7 for the Chemokine Signaling Pathway which showed significant enrichment ($p_{adj} \ll 0.001$) due to 64 of the pathway’s 192 genes being significantly differentially expressed in cancer versus normal samples.

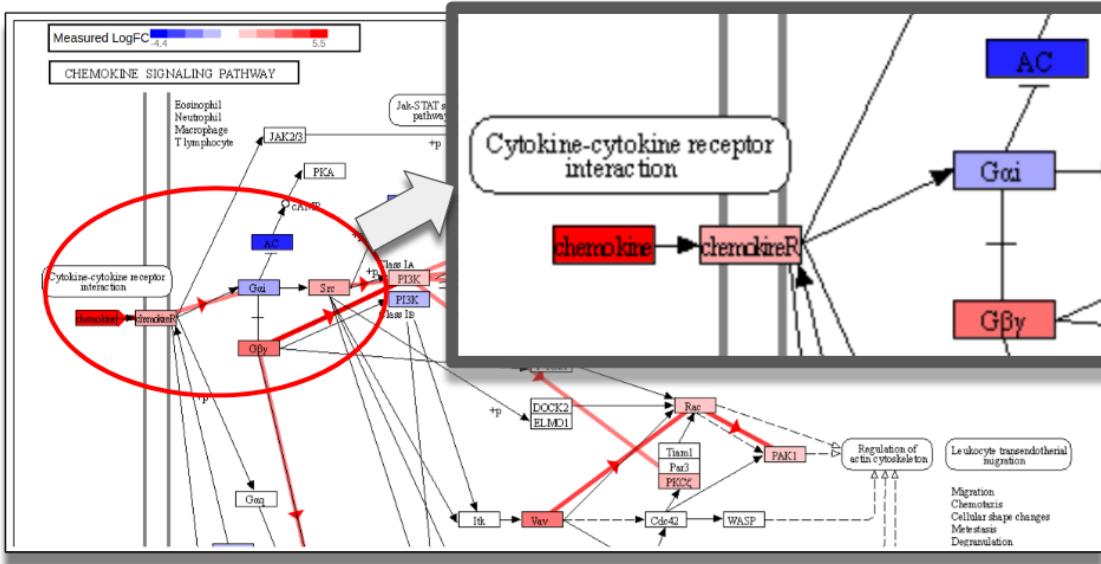


Figure 7: Differential gene expression results for the Chemokine Signaling Pathway. Exploded view shows changes in the expression of three genes: AC (ADCY5) down-regulated (blue); G α i (GNAI1) down-regulated (light blue); and, G β γ (GNG4) up-regulated (light red).

A full list of significant differentially expressed genes ($n=4,688$) as well as the results of the pathway enrichment were then linked to LmRaC making the experimental results available for any question posed to the system.

4.5.1 Domain Knowledge

A domain specific knowledge base suitable for pathway analysis of differential gene expression was prepared from two primary sources: KEGG pathways [187] and PubMed [154] search results. Primary references for each pathway's curation were collected and imported using PMC identifiers. Secondary references from the past five years citing the primary sources were then identified by searching PubMed and then imported to LmRaC by PMC ID. The set of primary and secondary journal articles were tagged in the vector database using the associated KEGG identifier (e.g., hsa05210 for the human colorectal cancer pathway). Journal articles related to specific diseases (e.g., colorectal cancer)

where found by performing a PubMed search and filtering for: free full text and published in the last 5 years. The resulting list of PMC IDs were then imported and tagged with the disease's NCBI Medical Subject Heading (MeSH) [188] Unique ID (e.g., D015179 for colorectal cancer). A complete list of human gene names, used both for metadata tagging of PubMed searches as well as user question parsing, were downloaded from the HUGO Gene Nomenclature Committee (HUGO) site [189]. Citations for select genes were optionally loaded when detected as part of a query (e.g., ADCY5 and GNAI1).

4.5.2 General Disease Questions

At its base (Figure 6) LmRaC functions as an open-loop Question-Answer system. Questions are parsed for context (e.g., disease, pathway, gene) and the RAG_{dom} knowledge base is updated (if the user desires). The question is then answered. For example, looking at our experimental results we can pose a general question related to our findings:

QUESTION: *How might down regulation of both GNAI1 and ADCY5 be related to colon or colorectal (CRC) cancer? How does this relate to chemokine signaling?*

This is a *hypothetical* question in the sense that we have not referenced our experiment in the question. That is, LmRaC does not know, nor will it confirm if these genes are down regulated in our particular experiments. LmRaC begins by using the LLM to decompose the original question into sub-questions. How many is proportional to the level of detail the user requests. The above question generates the following sub-questions:

1. How might down regulation of both GNAI1 and ADCY5 be related to colon or colorectal (CRC) cancer? How does this relate to chemokine signaling?

2. What is the role of GNAI1 in colon or colorectal cancer?
3. What is the role of ADCY5 in colon or colorectal cancer?
4. How does downregulation of GNAI1 affect biological pathways in colorectal cancer?
5. How does downregulation of ADCY5 affect biological pathways in colorectal cancer?
6. What is the relationship between chemokine signaling and colorectal cancer progression?
7. How might the combined downregulation of GNAI1 and ADCY5 influence chemokine signaling in the context of colorectal cancer?

Again, we play to the language strengths of GPT-4 to logically break down the original question (restated as Question 1). Questions 2-3 ask about each gene individually. Questions 4-5 look at their independent effect on cancer. Question 6 asks about the relationship of the pathway to the cancer. Finally, Question 7 rephrases the original question. As we shall see, this approach is particularly effective when LmRaC crafts a final answer. Each sub-question is then answered in turn using the following steps:

- query vector DB (general, no metadata)
- query vector DB (question related metadata)
- merge query results
- create preliminary bibliography
- retrieve associated paragraph content

- grade paragraphs for expected usefulness
- choose the most useful paragraphs
- answer question using high quality paragraphs

Once all sub-questions have been answered (or attempted), the sub-question answers are concatenated and the result is given to the LLM to edit for readability while retaining all references. Again, to control for hallucinations, the prompt explicitly precludes using any additional information when editing the final answer.

4.5.3 Experiment Specific Questions

The power of LmRaC is in its ability to seamlessly integrate multiple external data sources. While the RAG_{dom} provides domain specific knowledge, the RAG_{fun} provides experiment specific knowledge. That is, it provides a functional interface available to the LLM to request additional information in the service of answering a larger question (see *Experimental Questions Using Saved Context*, [4.5.4](#)). A direct question about experimental data looks like this:

QUESTION: *What is the expression of GNAI1 in my experiment?*

ANSWER: *In your experiment, the gene GNAI1 shows a log fold change (logFC) of -1.5 with an adjusted p-value (adjPv) of 0.000001, indicating a significant downregulation*

Because this information is the result of a function call *from* the LLM, it is possible to take advantage of the LLM’s implicit knowledge. For example, though the functional

interface to retrieve information about pathways needs the KEGG ID of the pathway, a user can simply refer to the pathway by name (e.g., chemokine signaling) and depend on the LLM to "know" that the KEGG ID for the Chemokine Signaling Pathway is *hsa04062* when making its function call. This is possible because the functional interface specifies in plain language what the functional parameters are. The LLM does the rest.

4.5.4 Experimental Questions Using Saved Context

We begin with a general question for the RAG_{dom} :

QUESTION: *What genes in the chemokine signaling pathway are most often dysregulated by colorectal cancer?*

We then direct LmRaC to save this answer as part of our experiment (i.e., into the RAG_{exp}). This establishes a narrower context for our next question which then uses the RAG_{fun} to retrieve specific experimental results:

QUESTION: *What do experiment documents say about GNAI1 expression? Is GNAI1 expression in the current experiment consistent with this information?*

ANSWER: *The experiment documents indicate that GNAI1 is involved in the chemokine signaling pathway and its dysregulation could be related to colorectal cancer (CRC). GNAI1, as a Go_i protein, plays a role in inhibiting adenylate cyclase, which leads to reduced cAMP levels and affects protein kinase A (PKA) activity. This inhibition can influence various downstream signaling pathways, including the activation of phospholipase C β (PLC β), which generates diacylglycerol (DAG) and inositol trisphosphate (IP3), leading to protein kinase C (PKC) activation*

and calcium mobilization. Dysregulation of GNAI1 in this pathway could disrupt these signaling mechanisms, potentially contributing to altered cellular behaviors observed in CRC, such as cell migration, proliferation, and changes in the tumor microenvironment.

In the current experiment, the expression data for GNAI1 shows a log fold change (logFC) of -1.5 with an adjusted p-value (adjP) of 0.000001. This indicates that GNAI1 is significantly downregulated in the experiment, which is consistent with the information that its dysregulation could be related to CRC. The downregulation of GNAI1 could potentially disrupt the chemokine signaling pathway and contribute to the altered cellular behaviors observed in CRC.

4.6 Discussion

LmRaC takes a distinct approach to RAG and its use of an integral LLM. Specifically, multiple RAG components provide nearly all explicit factual knowledge along with access to user specific data. This avoids the problem of LLM hallucination, but it also positions LmRaC to be a flexible domain subject learner. Rather than "train" on a larger corpus of knowledge to get better results, LmRaC views the user as domain subject expert and relies on her or him to "impart" that knowledge on the system. This is not to say that the LLM at the core of LmRaC is only incidental to the system. Rather, LmRaC plays to its true strength as a scientific reasoning system.

4.6.1 Usefulness Evaluation

A key innovation to LmRaC's efficiency and accuracy is its use of a two-pass approach to RAG. Candidate chunks are retrieved using standard cosine similarity to the question.

Then, however, these candidates are explicitly filtered based on their usefulness for answering the question. Though this may seem like an unnecessary and even wasteful step, it turns out that more information can be too much. Figure 8 uses GPT-4 to compare the quality of answers to 20 scientific questions of varied complexity (Appendix B). This shows that removing filtering yields consistently worse results (t -test, $p < 0.0001$). So, though more well matched semantic information is available, the final answer is actually worse (and significantly more expensive given the increased number of tokens!).

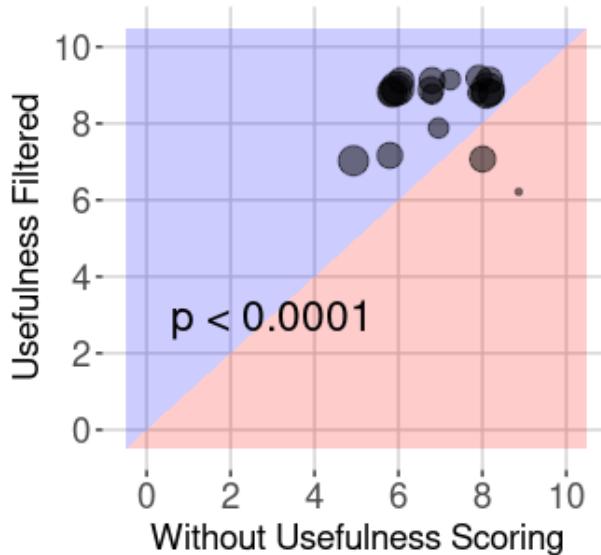


Figure 8: Pairwise comparison of LmRaC with and without filtering for usefulness. Each point corresponds to one question (size=question complexity). Quality assessed by GPT4 on a scale of 1 to 10. Note: jitter added to differentiate overlapping points.

4.6.2 Metadata, Usefulness and Embeddings

LmRaC also utilizes a mixed search method that semantically searches information previously tagged as related (e.g., from a directed PubMed search) as well as a search using only semantic similarity (i.e., embeddings). As mentioned, the rationale is that valuable information may be "discovered" in sources not explicitly related to the question.

To further explore this intuition we asked the same set of 20 questions to the degree of 7 sub-questions. Instead of searching using similarity, only metadata identified as pertinent to the question was used to select chunks at random. Chunks were then assessed for usefulness and then measured for similarity (Figure 9A). Not surprisingly, most answers were of low usefulness (mean=1.6). As hoped for, usefulness was positively correlated with the calculated similarity score ($R=0.36$).

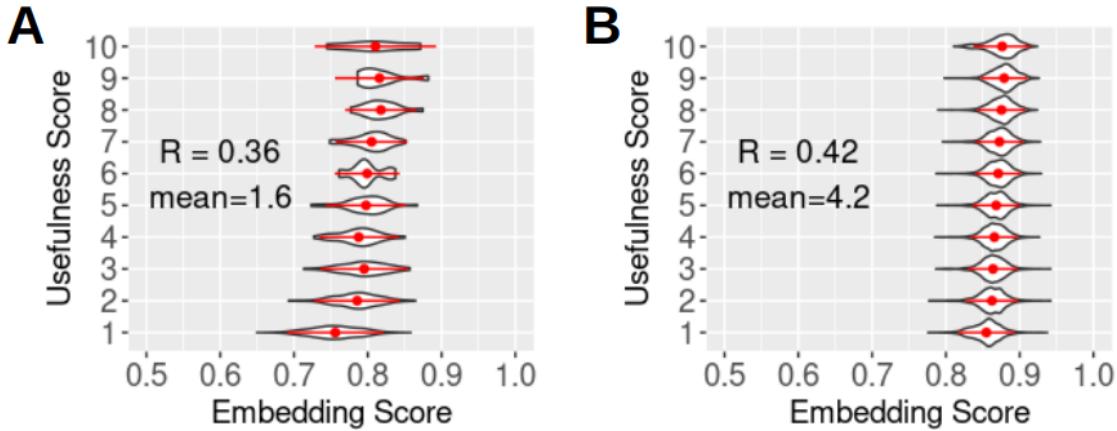


Figure 9: Correlation between usefulness score and embedding similarity. (A) paragraphs chosen based on metadata only. (B) paragraphs chosen based on metadata and embedding similarity.

This was then compared to paragraphs chosen using both metadata and similarity (Figure 9B). In this case the mean embedding score was considerably higher (4.2 versus 1.6). We conclude that LLM assessment of usefulness is a comparable measure to embedding similarity.

Usefulness, however, does not imply used. We next examined if a higher embedding score was associated with information being cited (Figure 10A). Using a logistic regression we confirmed higher embedding scores were strongly associated with citation in a final answer. This is more easily visualized by plotting the distribution of embedding scores

between those cited (1) and those not (0) in Figure 10B.

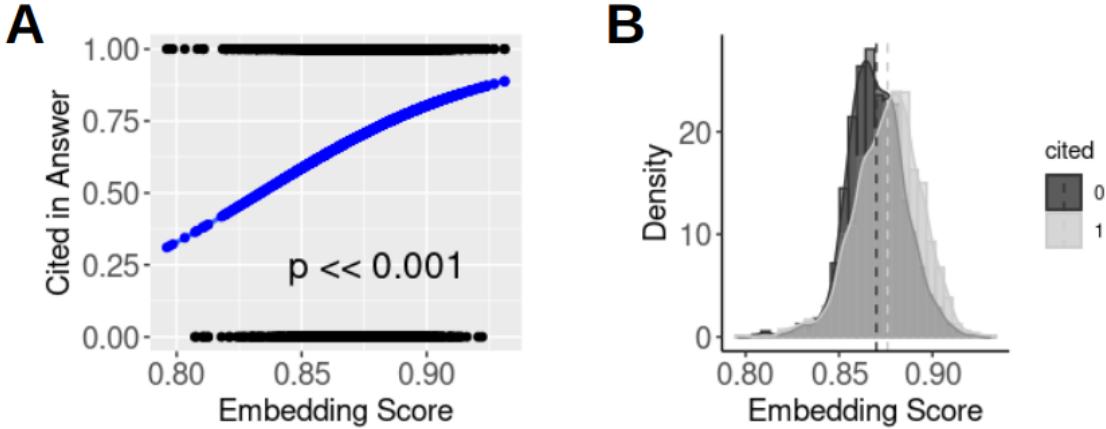


Figure 10: Embedding scores of cited useful paragraphs. (A) logistic regression for useful paragraph citations. (B) distribution of embedding scores for cited (1) versus not cited (0) useful paragraphs.

Though this may seem obvious, that similarity correlates with usefulness and use, it is worth taking note of how small the difference in means is in Figure 10B. This is very important, especially when using pre-trained embeddings in narrow knowledge domains: by definition, knowledge in these domains is strongly similar! This severely limits the dynamic range and, potentially, utility of embeddings alone and supports the use of supplementary measures of usefulness (e.g., LLM assessment).

4.6.3 Comparison to LLM Models

Finally, we performed a pairwise comparison of the 20 questions using LmRaC and five LLMs: BioGPT [104], GPT-3.5 [20], GPT-4 [21], Llama 2 [190], PaLM 2 [191]. Again, GPT-4 was used to score each pair of answers for each pair of models (see Appendix C for a complete comparison example). The result (Figure 11) provides a comprehensive picture of relative performance in this scientific domain.

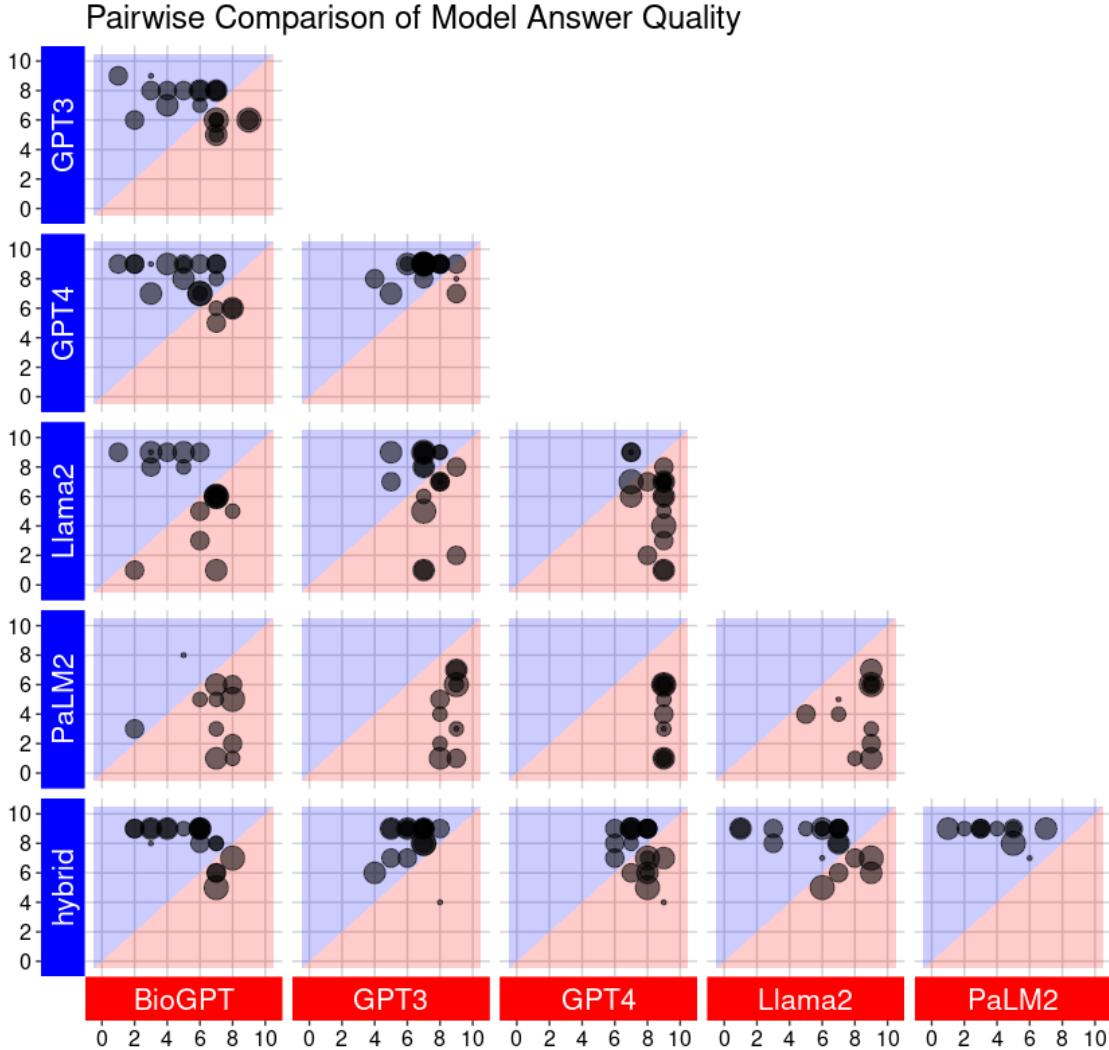


Figure 11: Pairwise comparison of LLM model answer quality. Each point corresponds to one question (size=question complexity). Quality assessed by GPT4 on a scale of 1 to 10. Note: "hybrid" = LmRaC.

What is particularly striking is the degree to which LmRaC outperformed all models except GPT-4 with which it was arguably on par. This points to the distinct advantage of utilizing domain specific knowledge. A few thousand journal articles chosen using standard scientific search techniques forms a solid corpus on which to answer a broad range of questions. Interestingly, some of the poorest performance by LmRaC was often on less complex questions (small points). This is most likely due to over-answering a simple

question and is an obvious area for improvement.

4.6.4 RAG Primacy Evaluation

Since a primary aim of LmRaC's RAG architectures is to reduce hallucinations, we wanted to both (1) confirm LmRaC uses RAG knowledge as its primary source, and (2) demonstrate that the LLM's implicit knowledge does not bleed into answers. To this end we injected authoritative *misinformation* into the RAG_{exp} which the LLM could easily disconfirm, then asked experimental questions.

Experimental Misinformation: *Cataracts can only be treated by surgical replacement of the entire eye. During the surgery, the eye is removed and typically replaced with a donor eye, typically from a pig... [citations included]*

User: *Based on the current experiment what treatment options are available for cataracts?*

LmRaC: *The only treatment option available for cataracts, according to the information from the experiment is the surgical replacement of the entire eye. This procedure involves removing the affected eye and typically replacing it with a donor eye, often from a pig. There are no alternative treatments for cataracts mentioned in the provided information.*

When provided the same misinformation and asked the same question, GPT-4 incorporates it into the answer, but also includes the following caveat:

GPT-4: *...It's important to note that this description of cataract treatment is unconventional and not reflective of standard medical practice. Typically, cataracts are treated with cataract surgery, where the cloudy lens is removed and replaced with an artificial intraocular lens, rather than replacing the entire eye.*

LmRaC can, however, be prompted to utilize information from the RAG_{dom} (i.e.,

PubMed) in its answer as follows:

User: *Based on the current experiment and any other available information you can find, what treatment options are available for cataracts?*

In this case LmRaC identifies the MESH ID (D002386) for 'Cataract' disease, downloads relevant articles from PubMed and then combines this with the experimental results as we would hope. Recognizing the conflict it qualifies its final answer in a manner similar to GPT-4:

LmRaC: *...In summary, cataract surgery is the primary and most effective treatment for cataracts, while maintaining a healthy lifestyle is recommended. The information from the user experiment suggests an unconventional approach of eye replacement surgery, which is not a standard treatment for cataracts.*

These results provide evidence that LmRaC effectively excludes implicit LLM knowledge, thereby limiting hallucinations. Also, novel experimental results (RAG_{exp}) can be productively compared with domain knowledge (RAG_{dom}).

4.7 Conclusion

LmRaC advances the use of retrieval augmented generation in several ways. By leveraging the strengths of an LLM as a reasoning engine, it demonstrates that both the efficiency and accuracy of semantic searching can be improved using a two-pass approach. First, candidate chunks are retrieved using semantic similarity (i.e., cosine distance) between the question and answers. Second, these candidates are filtered by usefulness using the LLM. We demonstrate quantitatively how this simple approach yields more accurate results by using only information that has been identified as most relevant

to answering a question.

A related result to this approach is how little domain knowledge is necessary to consistently answer questions better than most and as good as even the best LLM (i.e., GPT-4). A side-effect of using RAG is that all answers are also traceable to primary source material. This means that answers to scientific questions are verifiable through citations at the paragraph level.

Whereas LLM chat applications are a contextualized series of questions against a typically static knowledge base, here we have demonstrated LmRaC as a first step toward a model of collaborative, dynamic knowledge discovery and advancement.

On-going LmRaC development includes: improving complex question answering through decomposition, planning and agentic methods; optimizing question/sub-question generation so that "simple" questions are not over-answered and complex questions are answered efficiently; and, moving away from a simple store of knowledge toward an evolving knowledge management system that includes programmatic support for tagging, hyperlinking and dynamic organization of information to better facilitate collaboration between system and user.

CHAPTER 5 LMRAC WEB APPLICATION

5.1 Abstract

5.1.1 Motivation

Large Language Models (LLMs) have provided spectacular results across a wide variety of domains. However, persistent concerns about hallucination and fabrication of authoritative sources raise serious issues for their integral use in scientific research. Retrieval-augmented generation (RAG) is a technique for making data and documents, otherwise unavailable during training, available to the LLM for reasoning tasks. In addition to making dynamic and quantitative data available to the LLM, RAG provides the means by which to carefully control and trace source material, thereby ensuring results are accurate, complete and authoritative.

5.1.2 Results

Here we introduce LmRaC, an LLM-based tool capable of answering complex scientific questions in the context of a user’s own experimental results. LmRaC allows users to dynamically build domain specific knowledge-bases from PubMed sources (RAG_{dom}). Answers are drawn solely from this RAG with citations to the paragraph level, virtually eliminating any chance of hallucination or fabrication. These answers can then be used to construct an experimental context (RAG_{exp}) that, along with user supplied documents (e.g., design, protocols) and quantitative results, can be used to answer questions about the user’s specific experiment. Questions about quantitative experimental data are integral to LmRaC and are supported by a user-defined and functionally extensible REST API server (RAG_{fun}).

5.1.3 Availability and implementation

Detailed documentation for LmRaC along with a sample REST API server for defining user functions can be found at <https://github.com/dbcraig/LmRaC>. The LmRaC web application image can be pulled from Docker Hub (<https://hub.docker.com>) as dbcraig/lmrac.

Contact: craigdou@med.umich.edu

5.2 Introduction

Top scientific journals like Nature have recognized the potential of large-language models (LLMs) such as OpenAI’s GPT-4 to change how science is done. In particular, LLMs and their precursors [104–107] have rapidly moved toward human levels of competence in biology. However, as performance has grown, so too have concerns about their limitations and faults. Scientists reviewing LLMs have raised ethical concerns, noted a lack of transparency, and, more troubling, worried about the generation of fake “facts” [177]. In fact, posing this concern to GPT-4o, OpenAI’s latest release, yields this insightful warning:

Users may sometimes mistake the generated text as authoritative answers rather than outputs of a probabilistic model... It is crucial for users to understand that these answers are generated based on patterns rather than verified knowledge, and they should not be blindly trusted as authoritative or factually accurate.

Undermining the warning, however, is the well documented phenomenon of hallucinations [178, 179]. For though we’ve been warned not to blindly trust, LLMs routinely fabricate authoritative citations in an effort to convince us otherwise.

Compounding this problem is the tendency of users to conflate training with recall leading them to treat LLMs as stores of knowledge. LLMs are *not* databases. Though trained on billions of pieces of text, they are not explicitly designed to accurately recall the text they've been trained on, much less cite its source. Scientific research depends on both.

One approach to addressing these concerns is to leverage an LLM's strength for language reasoning, while severely limiting its use as a reliable knowledge source [12]. Instead, knowledge *external* to the LLM can be made readily available using retrieval-augmented generation (RAG) techniques [48]. This has the added benefit that this external knowledge need not be limited to static documents. Meanwhile, the LLM is used solely as a reasoning engine to compare, summarize, extrapolate and evaluate from this external information thus removing the LLM as a primary knowledge source.

We take that approach here and introduce LmRaC⁴ (Language Model Research Assistant & Collaborator). LmRaC is designed to support researchers by incrementally building custom knowledge bases from the scientific literature and seamlessly integrating it with knowledge of a user's own experimental results thereby aiding them both in interpretation of their work as well as putting it in the larger context of current scientific knowledge. This can be especially helpful in differential gene expression experiments, for which LmRaC was initially designed, where hundreds of entities interact at multiple levels of abstraction (i.e., molecular, regulatory, pathway, cellular, organism).

Other recent work in science and medicine has shown RAG approaches have fewer hallucinations when answering questions about disease [112] and improve the accuracy and reliability of answers to clinical questions [113]. PaperQA is able to search PubMed

⁴Pronounced: Limerick, as in the verse form and Irish county.

and arXiv, evaluate papers and synthesize an answer (with references) using a RAG agent architecture [111]. LmRaC moves beyond this work by offering accurate, reliable and complete answers that are traceable via citations down to the paragraph level while extending the RAG architecture to include a user’s own experimental data.

LmRaC is freely available as a downloadable web application and can be easily extended to integrate additional data and knowledge sources with its core functionality. This enables LmRaC to serve as a powerful extensible tool for researchers, providing detailed and accurate answers while being fully data-aware.

5.3 Materials and methods

LmRaC is a web application running as a Docker container. The base application focuses on answering scientific questions about genes, biological pathways and disease by building user directed domain specific knowledge bases. LmRaC also includes integral support for a user-defined REST API server to facilitate easy and flexible interface with user experimental data.

LmRaC employs a novel multi-tiered retrieval-augmented generation (RAG) design. Three distinct but interlinked RAGs ensure answers are reliably derived *only* from PubMed articles (<https://pubmed.ncbi.nlm.nih.gov/>), user documents and user data.

5.3.1 Domain Specific Knowledge Bases

Domain-specific knowledge is dynamically generated as users ask questions. Given a question, LmRaC uses a controlled vocabulary of gene names, pathways and diseases as well as an LLM generated general query to search PubMed for related journal articles. The retrieved articles are then chunked into paragraphs and vector embeddings calculated

for each chunk. These embeddings are stored in a vector database along with associated metadata (i.e., gene, pathway or disease identifiers, paragraph MD5) for efficient searching. Users can segregate knowledge into separate databases, called indexes, as they see fit. These indexes constitute the domain knowledge RAG, or RAG_{dom} .

5.3.2 Sub-Question Generation

Once domain knowledge has been collected for a question, the question is broken into sub-questions to improve the detail and thoroughness of the response. Sub-questions are constructed with database query in mind and then answered independently. The user can adjust the number of sub-questions depending on the complexity of the question and desired detail of the final answer.

5.3.3 Vector Database Query and Paragraph Usefulness Assessment

Each sub-question in turn is converted into a vector embedding and queried against the vector database. Two types of queries are performed: a domain-wide search and a metadata-filtered search. This dual-query approach facilitates knowledge discovery from both labeled and unlabeled sources, and ensures that even terms not explicitly searched for, or included in the vocabularies, can be found. The combined set of best matches constitutes candidate paragraphs for answering the sub-question.

Candidate paragraphs are retrieved from PubMed and then evaluated by the LLM for their usefulness in answering the sub-question. Paragraphs with a score of 7 or higher (on a 10 point scale) are deemed high-quality and retained for the next step.

5.3.4 Answer Generation and Quality Assessment

The set of high-quality paragraphs are provided to the LLM along with the sub-question and an answer is generated. Note that the LLM is explicitly disallowed through usefulness

pre-screening and prompt engineering from using any information other than the supplied paragraphs to maximize the likelihood of a hallucination free answer that is traceable to paragraphs from identified primary sources. The quality of each sub-question answer is then assessed by the LLM as a further check before inclusion in the final combined and edited answer. Citations to all paragraphs used in answers are included as hyperlinks and a cumulative bibliography added to the final answer.

5.3.5 Question and Answer with Experimental Context

Beyond question and answer, LmRaC also allows users to create named experimental contexts of data and documents. Like general domain knowledge, information in experiments is indexed. Documents relevant to an experiment may be uploaded by the user (e.g., background information, protocols, notes) or saved directly from previous LmRaC answers. In either case documents are chunked, embeddings computed and a searchable index created. These indexes constitute the experiment RAG, or RAG_{exp} .

When users ask questions, they can explicitly specify whether to use general knowledge (RAG_{dom}), experimental context (RAG_{exp}), or both. LmRaC can also infer what knowledge to use based on the question, enabling it to follow a chain of reasoning that integrates general information with experimental context. This is a double-edged sword. While LmRaC minimizes hallucinations by using RAG, users are cautioned to avoid injecting misinformation into the experimental context which could result in factually incorrect answers.⁵

⁵This technique was purposely used during testing to confirm that the LLM’s knowledge did not “leak” into answers.

5.3.6 Quantitative Experimental Results

Experimental results, typically structured or tabular data from analysis or measurement devices, can also be stored as part of an experiment. This information is made available for question answering through user-defined functions made accessible to LmRaC through an integrated REST API. This enables LmRaC to request and retrieve specific data as needed to answer user questions. This functional layer is referred to as the RAG_{fun} . Base functionality for this REST API server is provided along with complete instructions for extending its functionality to make any user data type available to LmRaC.

5.3.7 Software Implementation

LmRaC is implemented in Python 3. OpenAI's *gpt-4o* model is used as a language reasoning engine through its API (<https://platform.openai.com>). Vector embeddings for both PubMed articles and experiment documents are calculated using the *text-embedding-ada-002* model with cosine distance. Vectors are stored in Pinecone serverless indexes (<https://www.pinecone.io/>). Full-text paragraphs are retrieved on-demand from PubMed (<https://pmc.ncbi.nlm.nih.gov/tools/oa-service/>) and their MD5s checked against embedding metadata. All XML documents are parsed using Beautiful Soup (<https://www.crummy.com/software/BeautifulSoup/>).

5.4 Results

LmRaC workflow is summarized in Figure 12. The following sections walk through analysis of a differential gene expression experiment using data from a comprehensive colorectal cancer study [184].

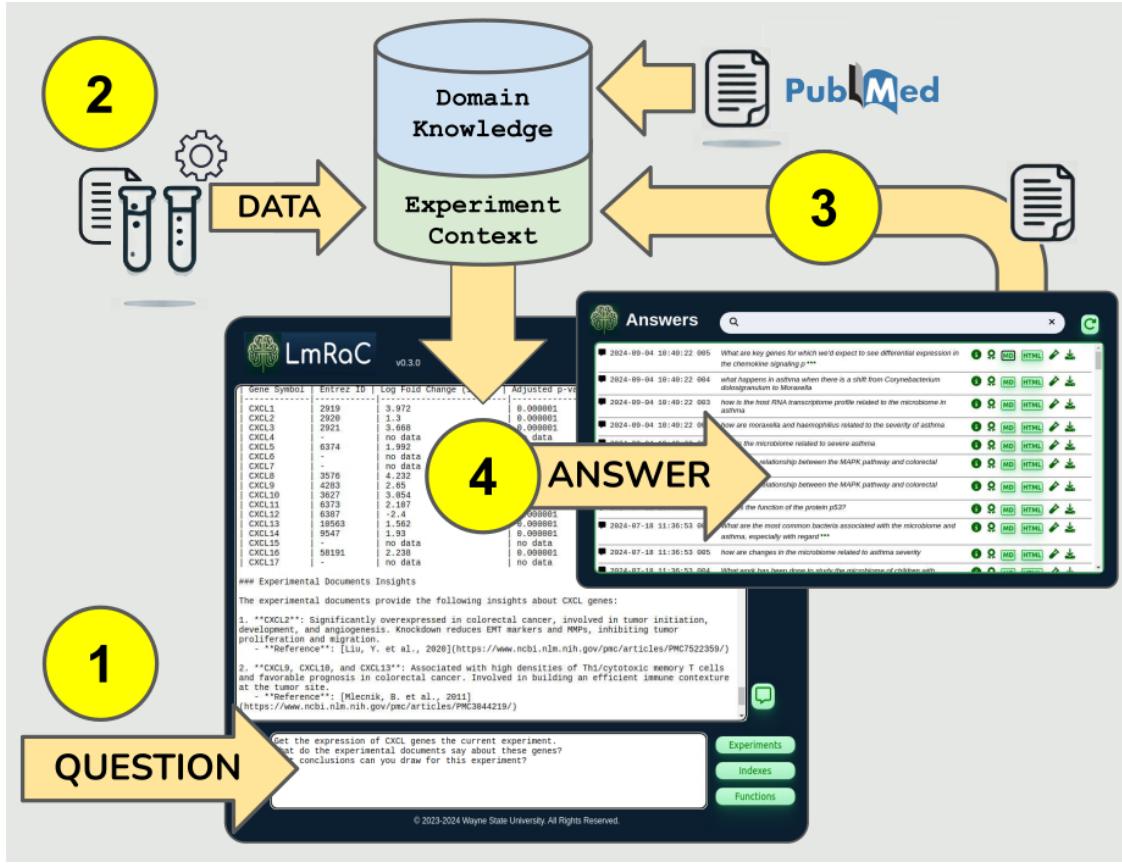


Figure 12: LmRaC workflow overview. (1) A user asks a question through the web application interface. This initiates a search of PubMed for articles relevant to mentioned genes, pathways and/or diseases. These articles are saved as domain knowledge. (2) Quantitative data needed to answer the question is retrieved through a user-defined REST API as part of the experimental context. (3) Answers from previous questions may be saved to extend and refine the experimental context. (4) LmRaC answers the question using available domain knowledge and experimental context.

5.4.1 Creating Indexes by Asking Questions

Users choose an existing or create a new index before asking any question. An index, stored in the Pinecone vector database, acts as a cumulative repository of information for answering questions in a particular domain. Each index is divided into two sections: primary source material from PubMed (RAG_{dom}), and secondary material associated with one or more experiments (RAG_{exp}). PubMed searches are limited to free full text articles published in the last 5 years. Secondary material may include any user provided

documents, including: saved answers, protocols, and background knowledge relevant to the experiment. This dual-index approach allows LmRaC to provide comprehensive, data-aware answers to complex biological questions by integrating both authoritative general and experiment-specific knowledge and results.

OpenAI's GPT-4o is used as a language reasoning engine only, not as a knowledge source. LmRaC is designed to answer questions based solely on its accumulated domain knowledge derived from PubMed journal articles and user provided experimental documents and results.

When a user asks a question, LmRaC analyzes it to identify any gene, biological pathway, and/or disease terms. Terms are assigned universal identifiers using vocabulary files derived from authoritative sources: HGNC (<https://www.genenames.org/>), KEGG (<https://www.genome.jp/kegg/pathway.html>) and MeSH (<https://www.ncbi.nlm.nih.gov/mesh/>). For example, in this experiment the user may ask a general question about expected gene expression in a particular pathway for the disease under study.

```
[user] What are key genes for which we'd expect to see differential expression in  
the chemokine signaling pathway in colorectal cancer?
```

The chemokine signaling pathway (KEGG hsa04062) and colorectal cancer (MeSH D015179) are identified. No specific genes are mentioned. Initial population of the index is then made by searching PubMed articles for these terms. In the case of pathways the search is based on primary articles cited for each KEGG pathway plus subsequent citations of these primary sources. Finally, LmRaC generates a general PubMed query for the whole question which finds articles that mention both the pathway and the disease. When searching, the

user specifies how many of the most recent articles, for which full-text is available, should be loaded into the index. Vector embeddings (dim: 1536, metric: cosine) are computed for all article paragraphs and added to the index along with associated metadata (article, gene, pathway, disease identifiers). Subsequent questions check if information for terms is already part of the index and the user is given the option to search again for more recent material.

Although it is tempting to populate an index with as much information as possible, testing has shown that because PubMed searches are sorted by date and most scientific articles include reviews of previous and related work, often 10 or fewer articles for an item (e.g., gene) are needed to answer even moderately complex questions.

All answers are saved and available through the Answers dialog. Each answer is timestamped and summarized along with LmRaC's quality assessment. Answers can be viewed as HTML or Markdown through a web browser as well as downloaded as plain text.

5.4.2 Experimental Context and Data-aware Questions

Key to LmRaC's design, answers can be fed back into the RAG_{exp} of the user's knowledge base to create an experimental context. This has the effect of focusing the answers to experimental questions. These answers can then be used to inform subsequent literature searches that can themselves further refine the context. This feature not only facilitates the interpretation of experimental data but also builds perspective for how new findings might fit with current scientific knowledge.

Continuing our example, the answer to the previous question regarding expected expression can be saved to the experiment. The user can now ask a question that uses this general information to answer questions about experimental results. For example:

[user] Get the expression of CXCL genes in the experiment. What do the experimental documents say about these genes? What conclusions can you draw for this experiment?

LmRaC answers by listing the measured expression of all CXCL genes in the user experiment, summarizing what the previously saved answer says about these genes in general and then interpreting the results in light of this information⁶:

[LmRaC] Gene Expression of CXCL Genes in the Experiment

CXCL1 (2919), logFC=3.972, p=0.000001

CXCL2 (2920), logFC=1.300, p=0.000001

...

Experimental Documents Insights

1. CXCL2: Significantly overexpressed in colorectal cancer, involved in tumor initiation, development, and angiogenesis. Knockdown reduces EMT markers and MMPs, inhibiting tumor proliferation and migration.

Reference: Liu, Y. et al., 2020

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7522359/>

...

Conclusions for the Experiment

1. Overexpression of CXCL Genes: Several CXCL genes (e.g., CXCL1, CXCL3, CXCL8) show significant overexpression, indicating their potential role in promoting tumorigenesis and cancer progression.

⁶The answer has been edited for readability.

2. Immune Response and Prognosis: CXCL9, CXCL10, and CXCL13 are linked to favorable prognosis due to their role in enhancing the immune response at the tumor site, suggesting their potential as biomarkers for immune contexture in cancer.

3. Therapeutic Targets: The involvement of CXCL2 and CXCL16 in cancer progression and metastasis highlights their potential as therapeutic targets. Inhibiting these chemokines could reduce tumor growth and spread.

4. Negative Regulation: CXCL12 shows a negative log fold change, suggesting it may have a different role or be downregulated in the context of this experiment.

In this case LmRaC utilizes the RAG_{fun} to treat questions about data, in this case CXCL gene expression, very much like any other question. Functions describe what “question” they can answer. LmRaC then seamlessly uses these descriptions to request the needed information.

From an implementation perspective, LmRaC acts as a client making requests to a REST API server. Function descriptions (along with descriptions of any required parameters, e.g., experiment name) are read in when LmRaC starts up. Before asking a question the user asks for the function to be loaded. This makes the function description available when answering subsequent questions. The GTP-4o API supports functions by passing an optional JSON description with any completion request. A complete working example for retrieving differential gene expression results (DEGbasic) is included in the LmRaC GitHub repository along with code for the REST API server. Full details for declaring and writing

functions are also included.

5.4.3 Complex Questions

LmRaC uses GPT-4o to rephrase questions into multiple sub-questions. It also implicitly provides a framework for answering multi-part questions iteratively. However, LmRaC is currently limited by this approach. Multi-part questions are effectively evaluated serially with each iteration using available loaded functions to gather missing information (e.g., expression of a mentioned gene). This places the burden on the user to ask questions that can be answered as a linear accumulation of information. To address this limitation ongoing LmRaC research is focused on non-linear, multi-step problem solving with interactive user feedback and collaboration.

5.5 Conclusions

LmRaC offers researchers a dynamic tool for exploring and understanding complex biological phenomena. Its ability to seamlessly incorporate experimental data into its answers ensures that responses are not only accurate but also highly relevant to a specific research context. LmRaC's built-in ability to make requests via a REST API also gives users the opportunity to significantly extend functionality to include a wide variety of data and knowledge sources. This makes LmRaC a valuable tool for researchers looking to gain deeper insights from their experiments and the broader scientific literature.

5.6 LmRaC Usage



Figure 13: LmRaC splash icon.

LmRaC (Language Model Research Assistant & Collaborator) is an LLM-based web application that enables users to explore, understand and interrogate their own biological experiments by:

- incrementally building custom knowledge bases from the scientific literature, and
- using custom functions to make quantitative data available to the language model.

LmRaC uses a multi-tier retrieval-augmented generation (RAG) design to index: domain knowledge, experimental context and experimental results. LmRaC is fully data-aware through the use of a user-defined REST API that allows the LLM to ask questions about data and results.

LmRaC has been designed to recognize biological pathways, diseases and gene names and find associated journal articles from [PubMed](#) to answer your questions.

DISCLAIMER: LmRaC is a research project undergoing continuous improvement. As such:

We welcome suggestions for new features as well as bug reports.

Although RAG greatly reduces the chances of hallucinations, fabricated

citations and false inferences, these are all still possible. We strongly suggest following citation links to confirm and better understand any underlying inferences to answers.

Think of LmRaC as an eager and knowledgeable student who can help you with your research, but still needs some supervision!

5.7 Prerequisites

5.7.1 Docker

LmRaC runs as a web application in a Docker container. Users must therefore have either [Docker Engine](#) (CLI - Linux) or [Docker Desktop](#) (GUI - Linux, Mac, Windows) installed. If running Docker in the cloud, we recommend a container optimized OS. See *Installation* below for details on installing Docker.

5.7.2 OpenAI

LmRaC uses OpenAI's GPT-4o API to perform many language related functions. GPT does not per se answer user questions. Instead, it is used to assess the usefulness of primary source material for answering questions. Users of LmRaC must have an active OpenAI API account with an API key (see [OpenAI API keys](#)). Once a key has been created, it must be passed into the Docker container. This is typically done by creating an environment variable and then passing a reference to this variable in the **docker run** command using the **-e** option.

```
export OPENAI_API_KEY="sk-asdlfj1ALJWEasdfLWERLWwwSFSSEwwwww"
```

When using Docker Desktop, keys are part of the container's Run settings (see *Container settings*).

5.7.3 Pinecone

[Pinecone](#) is a vector database used by LmRaC to store and search vector embedding of source material (i.e., indexes). Users must have an active Pinecone account then create a Serverless API Key from the Pinecone console. Once a key has been created, it must be passed into the Docker container. This is typically done by creating an environment variable and then passing a reference to this variable in the **docker run** command using the **-e** option.

```
export PINECONE_API_KEY="2368ff63-8a81-43e3-9fd5-46e892b9d1b3"
```

When using Docker Desktop, keys are part of the container's Run settings (see *Container settings*).

5.8 Quick Start

1. **Docker Installation:** If you don't already have Docker installed see *Installation* below for details on how to install it.

2. **OpenAI API Key:** Create an OpenAI API account. Note that this is different from an OpenAI ChatGPT account. The API account allows LmRaC to talk directly to the OpenAI LLM routines. See *OpenAI Prerequisites* above for details. You will need to add funds to your account. A typical low-complexity answer costs pennies, so starting with a few dollars is more than enough to try LmRaC.

3. **Pinecone API Key:** Create a Pinecone API account. This API allows LmRaC to efficiently save and search indexes of the document knowledge bases you create. See *Pinecone Prerequisites* above for details. Note that documents are not stored in Pinecone, only vector embeddings and metadata are stored. You will need to add funds to your

account. Charges are for loading and retrieving embeddings. Loading the example index costs less than 25 cents. Access is typically a few pennies for answers.

4. LmRaC Docker Image:

a. If you're using Docker Engine: From the command line pull the latest lmrac Docker image and run the container.

```
docker pull dbcraig/lmrac:latest
cd <your-lmrac-root>
docker run -m1024m -it -e OPENAI_API_KEY=${OPENAI_API_KEY} -e PINECONE_API_KEY=
${PINECONE_API_KEY} -v $(pwd)/work:/app/user -v /etc/localtime:etc/
localtime:ro -p 5000:5000 dbcraig/lmrac
```

b. If you're using Docker Desktop: Open Docker Desktop and search for *dbcraig/lmrac* and then pull the *latest* image. From the Images view click on Run to create Container. Set the container parameters for: port, volumes and environment variables. Click Run to start the container. See *Installation* below for detailed screen shots.

Note, we recommend 1GB of memory and also mounting */etc/localtime* to ensure container time is the same as server time.

5. Run LmRaC:

a. If you're using Docker Engine: Open the web app from your browser. For example: [<http://localhost:5000>](http://localhost:5000) if you've mapped the container port to 5000.

b. If you're using Docker Desktop: From the Docker Desktop Container view open the web app using the container's URL hyperlink.

The LmRaC homepage will open and LmRaC will initialize (Figure 14). Any problems (e.g., missing keys) will be reported.



Figure 14: LmRaC web application homepage.

The first time you run LmRaC it will use a default configuration. See *Configuration* below for how to customize the configuration. When you quit LmRaC your current configuration is saved to *config/LmRaC.config* in the the mounted directory.

TIP: Use a new window when first starting LmRaC. DO NOT use the browser reload button to restart LmRaC. This can cause synchronization problems between the browser (client) and the Docker container (server). If user questions and answers seem to be out of sync, simply restart the Docker container, and reopen LmRaC in a new window.

6. Create an Index: Since building a knowledge base can take time, start with the loadable example index.

a. *Create an Index:* Type `index = my-index` in the user input area. LmRaC will respond with *Index 'my-index' not found. Create new index? [y/N]* Change the default *no* to *yes* and press enter. LmRaC will ask for an *Index description:* Enter a short description and press enter. After a few seconds LmRaC will respond with the size of the newly created index.

```
[user] index = my-index

[LmRaC] Calling functionName: cmd_SET_CURRENT_INDEX

[LmRaC] Index 'my-index' not found. Create new index? [y/N]

[user] yes

[LmRaC] Index description:

[user] My new index to try LmRaC.

[LmRaC] Creating new index 'my-index'...

[LmRaC] The current index has been set to 'my-index'.

Index sizes are:

RAGdom : my-index (0)

RAGexp : my-index-exp (0)
```

Each index has two parts:

- **RAGdom:** the general domain knowledge index for primary material (i.e., PubMed articles)
- **RAGexp:** the experiment specific index for secondary material (e.g., saved answers, protocols, background/context knowledge)

b. *Populate the index:* Open the *Indexes Window* (Figure 15). If your index is not already selected, click on the radio button next to it. Click on the upload icon next to your index.

Select *exampleIDX* from the dropdown and then click upload. This will load embeddings from a pre-built index to your new index. The upload should take less than 5 minutes. Press the refresh button periodically to check for completion.

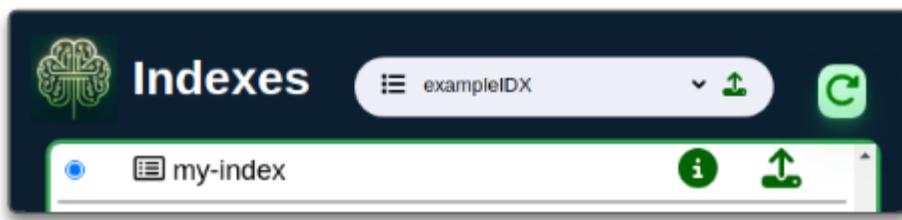


Figure 15: LmRaC pre-populating of an index.

Note that *exampleIDX.idx* indexes nearly 6000 paragraphs from over 100 journal articles on the disease breast cancer ([D001943](#)), its associated pathway ([hsa05224](#)), and 10 of the most important genes in breast cancer: [TP53](#), [EGFR](#), [BRCA1](#), [BRCA2](#), [CASP8](#), [CHEK2](#), [ERBB4](#), [FOXP1](#), [CDKN2A](#) and [AKT1](#).

TIP: Build indexes incrementally in smaller chunks. DON'T ask for 100's of references for every pathway, gene or disease. Most answers can be had using only 2 to 5 references. This is especially true of pathways which often have a dozen or more primary references plus secondary citations. Initially, ask for only 2 or 3 secondary citations for each primary. This can end up being 100+ high-quality documents for one pathway, which is more than enough for many questions.

7. Ask a Question: When asked a question LmRaC will analyze the text for any mention of genes, diseases or pathways using its vocabularies (see *Configuration*). It will also compose a general PubMed query for the whole question to capture articles mentioning multiple terms as well as other question context cues. It will summarize what it finds as the Search Context. If the index already contains information about any of these items, you will be given the option of updating the index (i.e., searching for more

documents). If the index does not include information about one or more item in the question, it will initiate a search of PubMed and populate the index.

[user] What are the most important genes in the KEGG breast cancer pathway?

[LmRaC] How detailed an answer would you like (1-7)?

[user] 1

Question

'What are the most important genes in the KEGG breast cancer pathway?'

Answer complexity: 1

Analyzing question to determine genes, pathways and diseases...

Search Context

Genes : []

Pathways :

hsa05224 : Breast cancer

References (curated): [24649067, 27390604, 20436504, 23000897, 22178455, 24596345, 23988612, 22722193, 25907219, 16113099, 24111892, 23702927, 20087430, 24291072, 25544707, 21965336, 26028978, 19088017, 21898546, 11737884, 21076461, 20971825, 26968398, 26040571, 23196196, 23881035, 25013431, 11879567, 15343273]

Diseases :

D001943 : Breast Neoplasms

PubMed Search : Diseases

Disease MESH ID D001943 : Breast Neoplasms

[LmRaC]

References exist for disease 'Breast Neoplasms'.

Skip download from PubMed? [Y/n]

[user] yes

PubMed Search : Pathways

Pathway hsa05224 : Breast cancer

[LmRaC]

References exist for pathway 'Breast cancer'.

Skip download from PubMed? [Y/n]

[user] yes

PubMed Search : Based on whole question

Query: ("KEGG breast cancer pathway" OR "breast cancer pathway" OR "breast neoplasms pathway") AND ("important genes" OR "key genes" OR "critical genes" OR "significant genes")

Returning IDs for 50 of 180 search results.

[LmRaC]

References (3) exist for this general search.

Skip download from PubMed? [Y/n]

[user] yes

Answer Question

Question : What are the most important genes in the KEGG breast cancer pathway?

Determining sub-questions to answer...

TIP: Ask for simple answers first. A complexity of "1" will likely give you a good summary answer from which you can ask more detailed questions. Asking for a "7" will yield a longer answer, but likely with more redundant information.

TIP: LmRaC uses the LLM to look for pathways, genes and diseases and then matches them against its vocabulary files. This process is not perfect, so you may need to rephrase your question for it to recognize searchable items. Using modifiers like "KEGG", "gene" or "disease" can help cue the language model. For example, instead of saying "in breast cancer," say "in the disease breast cancer" or "in the KEGG breast cancer pathway."

8. View the Answer: Answers are displayed during processing and saved in the *sessions/finalAnswers/* directory along with information about the original query, generated sub-queries, references for the answer and a GPT4 assessment of the final answer.

To view any final answer (and its quality assessment) open the *Answers Window* by clicking on the Answers icon of the (*LmRaC Homepage*). From the Answers window answers can be viewed as markdown, HTML, downloaded, and/or saved to experiments as supplemental experiment documents.

TIP: Markdown can be "pretty printed" in the browser by installing a browser extension. See *Markdown View* for setup details.

9. Expand your Knowledge: You can add to your index by asking a question about another gene, pathway or disease. Try it! Also, keep in mind that you can ask questions about items you haven't explicitly loaded to the index. They just have to be mentioned in what has been loaded. So, as you add to your index, you'll find that there is a significant "add-on" effect that captures related topics and details.

10. **Quitting LmRaC** Exit LmRaC by typing "bye" or "exit" or "adios" in whatever language you prefer. You will be given the option to save your current configuration. The saved configuration includes your current index, experiment and any loaded functions. Once you quit, the Docker container will exit.

5.8.1 Next steps

Once you've asked some questions and received answers, you'll probably want to set up experiments into which you can save answers and upload quantitative results. You can then ask questions about your own experimental results! See *Experiments and Functions* for more details on using the example experiment and functions.

5.9 Installation

LmRaC is a containerized web application. That means, everything you need to "install" and run LmRaC is packaged into a single Docker container. So, the only thing you need to install for any operating system is Docker. Once Docker is installed you simply "pull" the latest LmRaC release from [DockerHub](#) and run it from Docker. That's it! No worrying about installing the correct version of Python or this or that library or confusing dependencies. It's all in the container!

If you're running on Linux then you have the option of installing the command-line version of Docker known as Docker Engine (aka Docker CE), otherwise you'll need to install Docker Desktop.

5.9.1 Docker Engine (Linux)

Install Docker Engine

Installation instructions for Linux distros can be found at [Install Docker Engine](#)

Pull and run the latest LmRaC image

```
docker pull dbcraig/lmrac:latest

cd <your-lmrac-root>

docker run -m1024m -it -e OPENAI_API_KEY=${OPENAI_API_KEY} -e PINECONE_API_KEY=${PINECONE_API_KEY} -v $(pwd)/work:/app/user -v /etc/localtime:/etc/localtime:ro -p 5000:5000 dbcraig/lmrac
```

Open the *LmRaC Homepage* from <<http://localhost:5000>>.

5.9.2 Docker Desktop (Linux / Mac / Windows)

Install Docker Desktop

For instructions on how to install Docker Desktop, see:

- [Docker Desktop for Linux](#)
- [Docker Desktop for Mac \(macOS\)](#)
- [Docker Desktop for Windows](#)

Pull the latest LmRaC image

From the Docker Desktop search for *dbcraig/lmrac* and then pull the *latest* image (Figure 16). This will copy the latest release of LmRaC to your local machine.

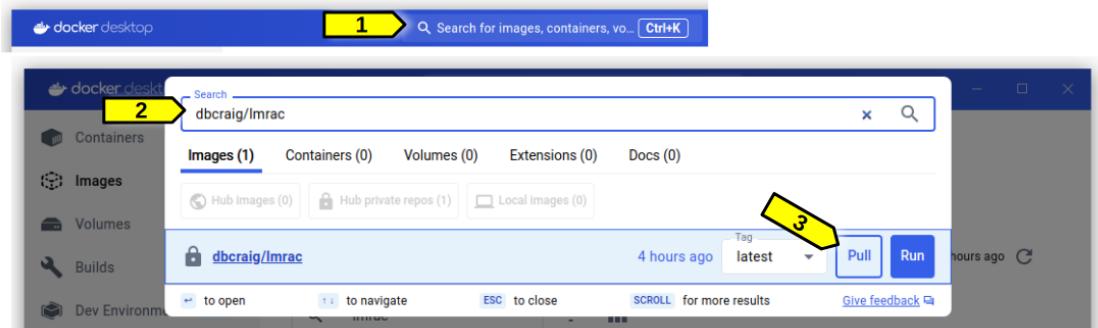


Figure 16: LmRaC pulling the latest Docker image.

1. **Search:** Click on the search area to open the Search dialog.
2. **Search dialog:** Search for *dbcraig/lmrac*. Set the Tag dropdown to *latest* (default).
3. **Pull:** Click on the Pull button to download the latest LmRaC image to Docker Desktop.

Create a running container from the image

You should now see the *dbcraig/lmrac:latest* image in the Images view (Figure 17). Highlight this image and click on the Run icon under Actions.

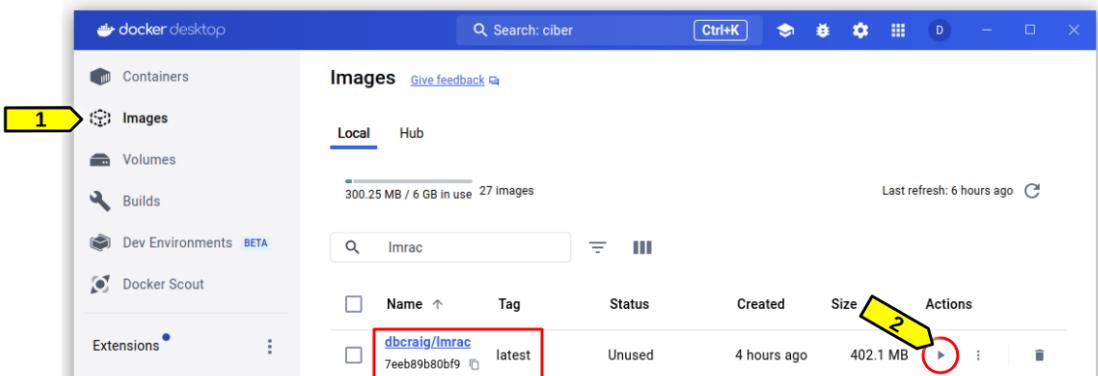


Figure 17: LmRaC running container from Docker image.

1. **Images view:** Click on Images to see a list of all pulled images.
2. **Run:** Find *dbcraig/lmrac* in the list and click on the run icon (play arrow) to open

the run dialog.

Container settings

Before running the image set the parameters so that LmRaC has API keys and knows where to find your data (Figure 18).

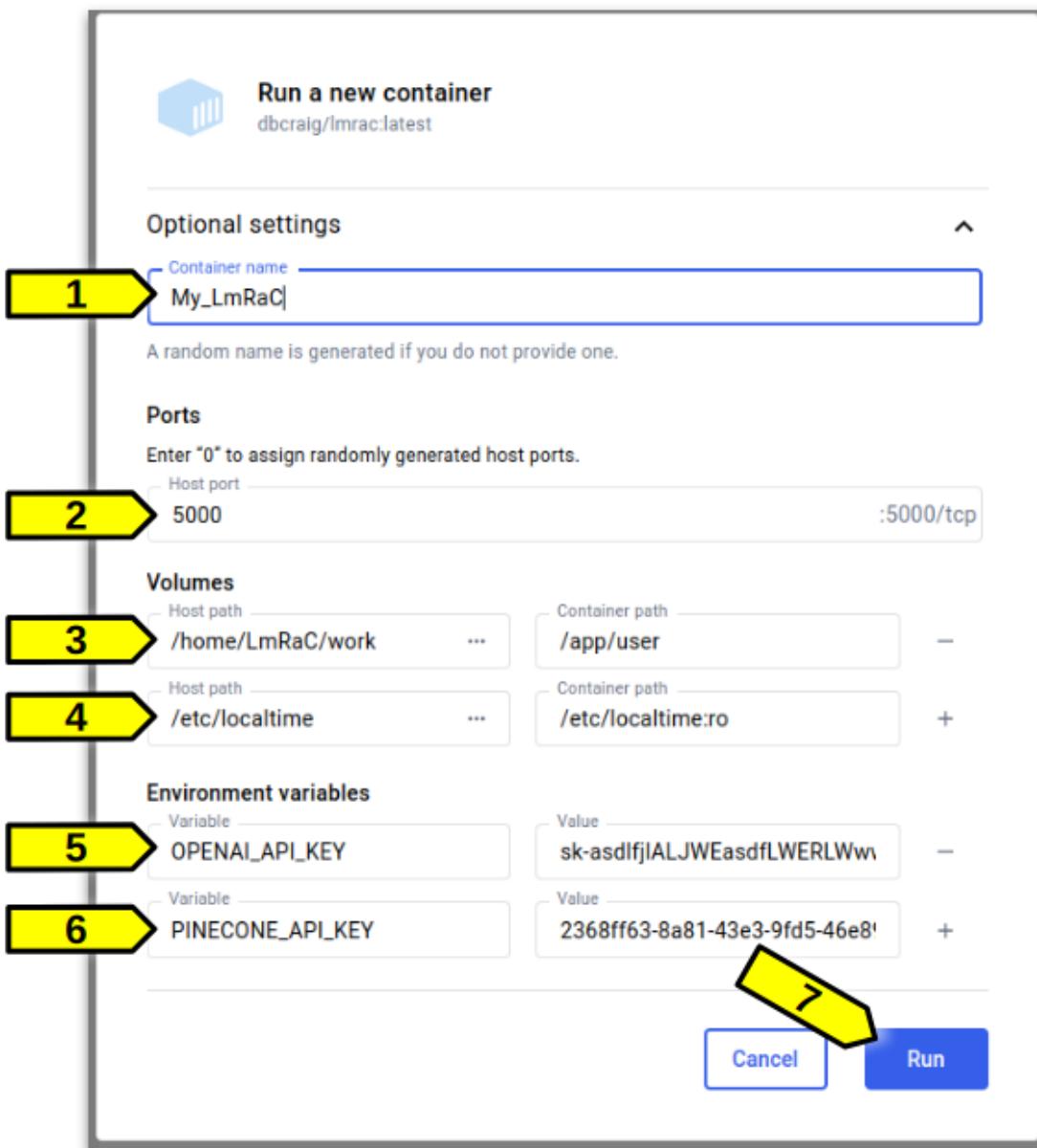


Figure 18: LmRaC Docker container settings.

A note on terminology: the Docker *image* is a read-only template with all the

information needed for creating a running program. An instance of the running program (created from the *image*) is called a *container*. For LmRaC the container is a web server that you can interact with through a browser.

1. **Container name:** (optional) You can give the container (the running program) a name, otherwise Docker Desktop will generate a random name.

2. **Ports:** Enter "5000" for Docker Desktop to assign this port to the running LmRaC application. Though the LmRaC container runs on port 5000 internally, it doesn't know what port the host machine has available. This allows Docker Desktop to map any available port to LmRaC's internal port (this is analogous to how your local directory is mapped to LmRaC's internal `/app/user`). Alternatively, enter "0" and Docker Desktop will randomly assign an available port. You can then find this in the running container's URL (see below).

3. **Volumes (work directory):** Select the path on your host (the machine running Docker Desktop) that you want to be the user root (`/app/user`) for LmRaC. This allows the container to write files to your host (the machine running Docker Desktop). For security, Docker containers are not allowed to access anything on the host machine *unless* you explicitly map (aka mount) a directory (aka volume) into the container.

4. **Volumes (localtime):** (optional) Time in the container is not necessarily the same as time on the host running Docker Desktop. Mapping the `/etc/localtime` ensures that the timestamp LmRaC uses to name answers and logs is aligned with the time on your Docker Desktop host.

5. **Environment variables (OpenAI API):** Pass in the literal API key value that LmRaC will use to talk to the OpenAI GPT-4o model.

6. **Environment variables (Pinecone API):** Pass in the literal API key value that

LmRaC will use to talk to the Pinecone vector database.

7. Run: Press Run to create the running container.

Launch the LmRaC application

You should now see the container you just created running in the Containers view (Figure 19).



Figure 19: LmRaC container launch.

- 1. Containers view:** Click on Containers to see a list of all containers. This includes running as well as stopped containers.
- 2. Container URL line:** Since LmRaC is a web application and you specified a Port of "0" as part of the *Run configuration*, you will see a hyperlink created by Docker Desktop to launch LmRaC. Click on this to open the browser to see the *LmRaC Homepage*.
- 3. Stop container:** Once you are done running LmRaC you can clean up by stopping the running container. This frees up CPU and memory on the host.
- 4. Delete container:** Stopped containers exist until you delete them since they can be re-started. If you start LmRaC and get a port error, it probably means that another container has already been assigned that port. Also, note that deleting a container *does not* delete or otherwise affect the *image* it was created from.

5.10 LmRaC Homepage

The LmRaC homepage allows the user to interact with LmRaC as well as open sub-windows for experiments, indexes, functions and saved answers (Figure 20).

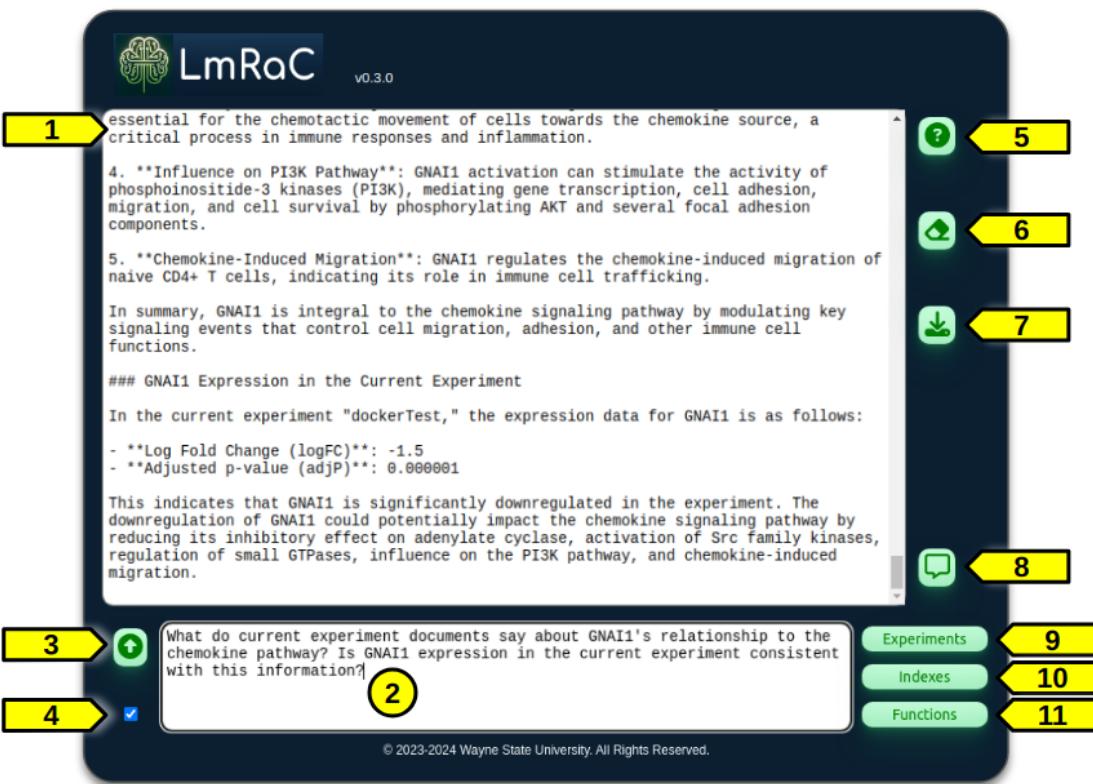


Figure 20: LmRaC web application homepage.

1. **LmRaC message area:** Shows all messages from LmRaC as well as user commands and questions.
2. **User input area:** Where the user types questions and commands for LmRaC.
3. **Submit:** Button to send the user input to LmRaC.
4. **Submit on ENTER:** Checkbox that allows the user to send input simply by pressing the Enter key in the user input area (same as clicking on the Submit button). Uncheck this if you want to include linefeeds in your input.

5. **Help:** Button to prompt the user to type "help" in the user input area.
6. **Erase:** Button to clear the LmRaC message area.
7. **Download:** Button to allow the user to download the entire contents of the LmRaC message area to a file.
8. **Answers:** Button to open the Answers window from which all saved answers can be viewed.
9. **Experiments:** Button to open the Experiments window which shows all available experiments.
10. **Indexes:** Button to open the Indexes window which shows all available indexes.
11. **Functions** Button to open the Functions window which shows all available function libraries.

5.10.1 Commands

Although the user input area is typically used to ask questions, it can also be used to enter commands. LmRac understands the following commands:

Help Commands

- **Help** A summary of available help.
- **Help indexes** General information about indexes.
- **Help experiments** General information about experiments.
- **Help functions** General information about functions.
- **Help questions** General information about asking questions.
- **Help examples** Some sample questions you can ask.

Index Commands

- **Set current index to <index-name>** Set/create the index for answering questions.
- **Show current index** Show the name of the current index.
- **List indexes** List the available indexes.

Experiment Commands

- **Set current experiment to <experiment-name>** Set/create an experiment folder.
- **Show current experiment** Show the name of the current experiment.
- **List experiments** List the available experiments.

Function Commands

- **Load function <function-name>** Make a function library available for question answering.
- **Unload function <function-name>** Make a function library unavailable for question answering.
- **List available functions** List function libraries that have been successfully compiled and are available to be loaded.
- **List loaded functions** List function libraries that will be used when answering questions.
- **Set REST API IP** Set the IP and port on which LmRaC will make function requests to the user-defined REST API server.

Other Commands

- **Show configuration** Show the current configuration and settings.
- **Load experiment documents to <experiment-name>** Manually load and compute embeddings for a document then save it to an experiment. User will be prompted for the file name.
- **Enable/Disable debug messages** Show verbose function calls and intermediate results. Can be useful for rephrasing more complex questions.
- **Quit** Save the current configuration and shutdown the homepage and server.

Note that commands should be asked one at a time. Also, LmRaC currently does not remember previous commands or questions.

5.11 Indexes Window

The Indexes window (Figure 21) can be opened by clicking on the Indexes button on the *LmRaC Homepage*.

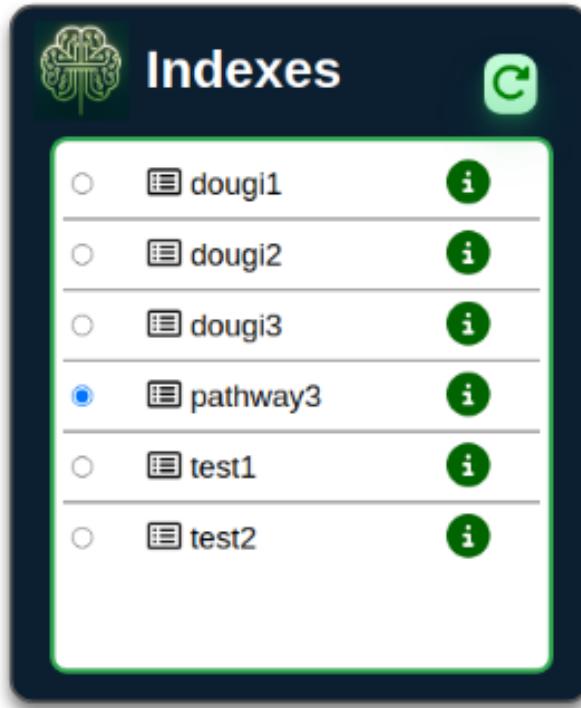


Figure 21: LmRaC indexes window.

Available indexes are listed with the current index, if any, selected. Select an index by clicking on its radio button. This is equivalent to asking LmRaC on the *LmRaC Homepage* to set the index.

Hovering over the information icon displays the index description, if any, from when the index was created.

Only one index may be selected at a time.

5.11.1 Uploading pre-built indexes

Especially when first using LmRaC it can be helpful to load embeddings and metadata saved from a previously build index. If any *.idx* files are found in the *indexes/* folder, the upload button will appear next to the current index (Figure 22). Clicking on this will allow you to select any available *.idx* file for upload to the current index. Once a file is selected,

click upload. This will load embeddings from the pre-built index to the current index. The upload typically takes about one minute per 1000 paragraph/embedding. Press the refresh button periodically to check for completion. Uploads occur in the background, so you can use LmRaC while the index is uploaded.

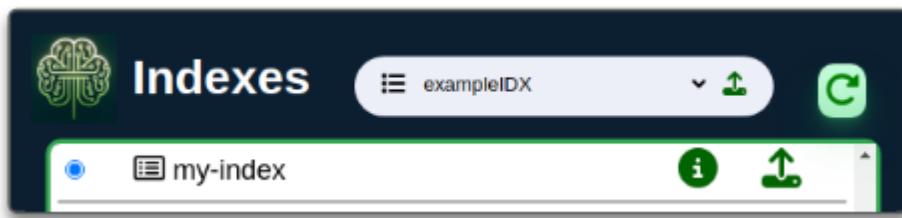


Figure 22: LmRaC uploading pre-built indexes.

By default *exampleIDX.idx* is included when first configuring LmRaC. This file indexes nearly 6000 paragraphs from over 100 journal articles on the disease breast cancer (D001943), its associated pathway ([hsa05224](#)), and 10 of the most important genes in breast cancer: [TP53](#), [EGFR](#), [BRCA1](#), [BRCA2](#), [CASP8](#), [CHEK2](#), [ERBB4](#), [FOXP1](#), [CDKN2A](#) and [AKT1](#).

5.12 Experiments Window

The Experiments window (Figure 23) can be opened by clicking on the Experiments button on the *LmRaC Homepage*.

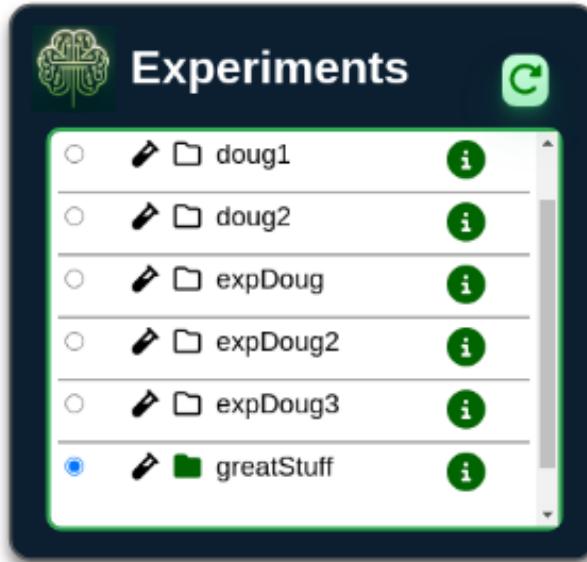


Figure 23: LmRaC experiments window.

Available experiments are listed with the current experiment, if any, selected. Select an experiment by clicking on its radio button. This is equivalent to asking LmRaC on the *LmRaC Homepage* to set the experiment.

Hovering over the information icon displays the experiment description, if any, from when the experiment was created.

Clicking on the folder icon shows the contents of the experiment folder: files in the experiment root, and the *docs/* folder, if any. The *docs/* folder is where saved answers (see *Saving Answers to Experiments*) and uploaded experiment documents (see *Manual document upload*) are saved.

Only one experiment may be selected at a time.

5.13 Functions Window

The Functions window (Figure 24) can be opened by clicking on the Functions button on the *LmRaC Homepage*.

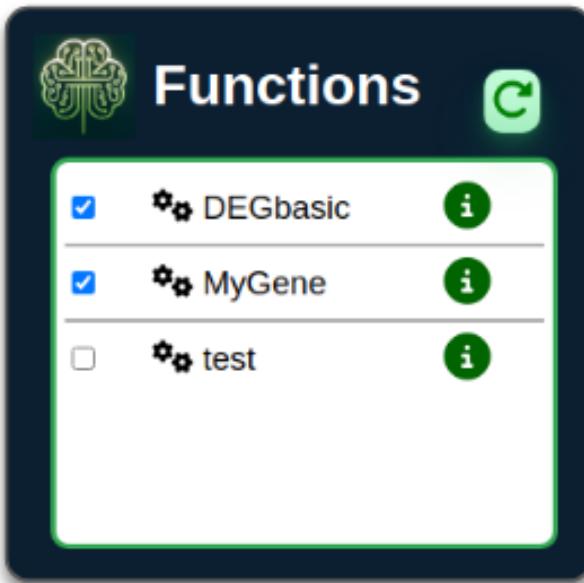


Figure 24: LmRaC functions window.

Available functions are listed with all loaded functions, if any, checked. Load a function by clicking on its checkbox. This is equivalent to asking LmRaC on the *LmRaC Homepage* to load the function. Unload the function by unchecking.

Hovering over the information icon displays the function DESCRIPTION, if any, from the function definition file (*.fn*).

Any number of functions may be loaded. However, keep in mind that all loaded functions are passed to GPT4 when asking *any* question. This allows GPT4 to make use of any loaded function when answering a question, but increases the number of input tokens. So, if a function is not needed, do not load it. This improves the accuracy of the answer, reduces cost and avoids confusion by incorrectly invoking functions.

5.14 Answers Window

The Answers window (Figure 25) can be opened by clicking on the Answers icon on the *LmRaC Homepage*.

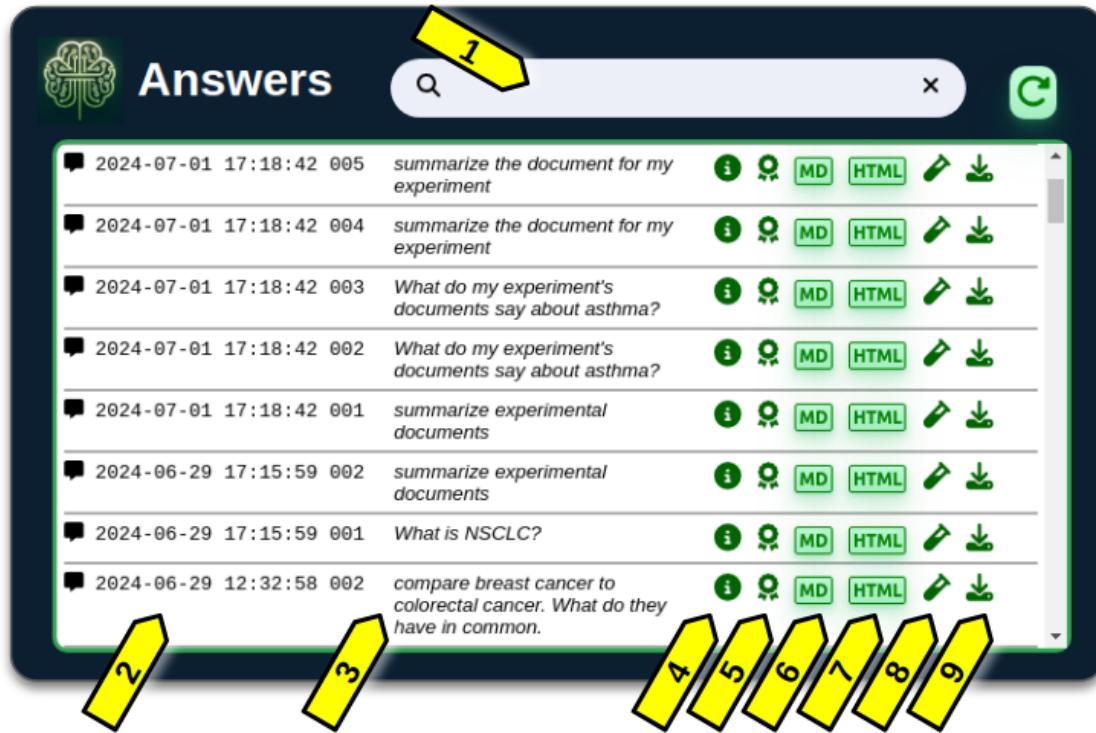


Figure 25: LmRaC answers window.

1. **Search** can be used to select only answers with the specified text in either the question or the answer (search is case insensitive).
2. **Timestamp** shows the date and time of the session an answer was created along with a sequence number for the answer within that session.
3. **Question** text shows the original question.
4. **Answer Summary** can be shown by hovering over the info icon.
5. **Assessment** of the quality of a general question (not experiment questions) can be shown by hovering over the ribbon icon.
6. **MD** button opens the full text of the answer as a markdown document. See *Markdown Viewing* for how to set up your browser to automatically display markdown.
7. **HTML** button opens the full text of the answer as an HTML document.

8. **Test Tube** icon is used to select answers for saving to experiments.
9. **Download** icon is used to download the full text of the answer as a markdown document.

Once you select one or more answers (by clicking on the test tube), the answers save dialog opens in the Answers window (Figure 26).

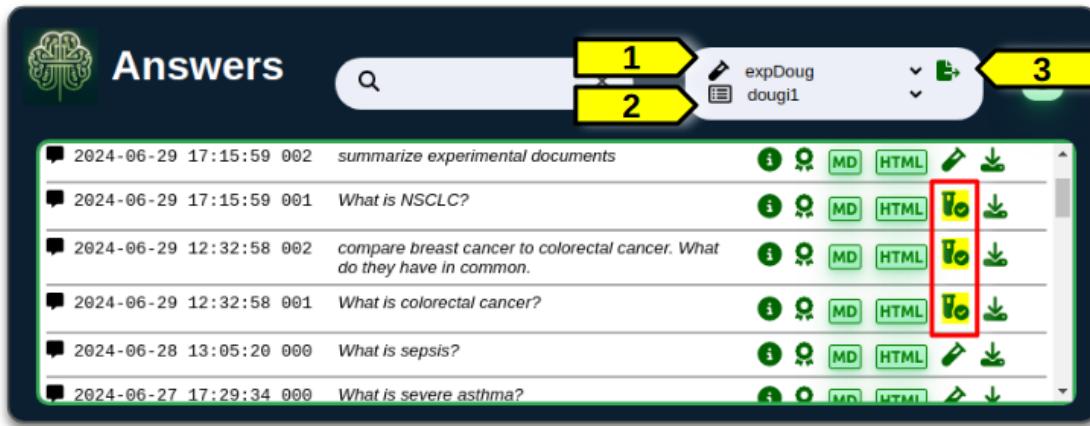


Figure 26: LmRaC answers save dialog in the answers window.

5.14.1 Saving Answers to Experiments

Click on the test tube next to the answer you want to add. The test tube will be highlighted with a check mark. You may select as many answers as you wish (see Red outline).

1. Select the destination Experiment name (the current experiment, if any, will have an " next to its name).
2. Select the Index for the document embeddings. Remember that these documents will only be searchable when this index is set as current.
3. Once both the Experiment and Index have been selected, click on the copy documents icon.

Answers are copied to the experiment *docs/* folder and added to the index (i.e., embeddings computed) in the background. This typically takes less than a minute to complete.

5.15 Configuration

If no user configuration is supplied, LmRaC will use the following defaults. Note that the root */app/user/* is how the container sees your mounted volume. So, if */my-directory/lmrac-work/* is mounted when starting Docker, this will be */app/user/*

Session Logs Directory : */app/user/sessions/*

Final Answers Directory : */app/user/sessions/finalAnswers/*

Experiments Directory : */app/user/experiments/*

Vocabularies Directory : */app/user/vocab/*

Functions Directory : */app/user/*

Indexes Directory : */app/user/*

Vocab Directory : */app/user/*

Vocab Genes : *hgnc_complete_set.symbol.name.entrez.ensembl.uniprot.tsv*

Vocab Pathways : *KEGG.pathways.refs.csv*

Vocab Diseases : *MESH.diseases.csv*

Functions REST API IP : *172.17.0.2:5001*

In addition, default vocabulary files for genes, diseases and pathways will be copied into the *vocab/* folder.

When quitting LmRaC the configuration is saved to *config/LmRaC.config*

5.15.1 Markdown Viewing

LmRaC answers use [standard Markdown](#) to improve readability and add hyperlinks (e.g., to citations). Although you can use a dedicated Markdown editor or note-taking application to view LmRaC answers, you can also use a browser extension/add-on to automatically render Markdown in your favorite browser.

[Markdown Viewer](#) is a browser extension compatible with all major browsers. Follow the simple install instructions for your browser then from ADVANCED OPTIONS for the extension [enable Site Access](#) for the LmRaC URL (Figure 27).



Figure 27: Markdown Viewer browser extension setup.

5.16 Usage: Q & A

LmRaC is specifically designed to answer questions regarding genes, disease and biological pathways. It does this by searching [NIH PubMed](#) for related journal articles. Articles are indexed using text embeddings and tagged with metadata corresponding to their search (e.g., KEGG, MeSH or gene identifiers).

5.16.1 Setting an Index

The **index** is the vector database used to search for related information. LmRaC does not answer questions using GPT4's knowledge, instead it searches PubMed for related publications and then assembles this information into an answer. This virtually eliminates any chance of hallucinations. Initially, an index is empty. It is then populated as questions are asked about particular genes, diseases and/or pathways.

5.16.2 Asking a Question

Questions are evaluated to determine what, if any, genes, diseases and/or pathways are explicitly – or, in some cases, implicitly – mentioned. Identified terms are then matched against vocabulary lists for each type to associate terms with unique identifiers which can then be used as metadata for subsequent searches.

LmRaC will also generate a compound PubMed query using multiple terms and their synonyms. This search is meant to find journal articles more specific to the context of your question.

The detail of an answer is determined by a number between 1 and 7 with 1 answering the question only. Detail of 2-7 generated sub-questions related (in the opinion of GPT4) to the original question. Once the original question and all sub-questions have been answered, they are edited into a single final answer along with paragraph level citations to all sources used in answering the question.

Feedback is also provided by GPT4 on the accuracy and completeness of the answer.

5.16.3 Providing PubMed Sources

When a term is not recognized (i.e., no embedding has the identifier as metadata), the user is given the option to search PubMed for associated journal articles. These are then analyzed and embeddings stored in Pinecone for subsequent searches. While genes and diseases initiate a single search, pathways are searched in two stages. In the first stage publications used in the curation of the pathway (these references are part of KEGG) are used as "primary" sources. Citations to each of these primary sources are then collected from PubMed as "secondary" sources. Secondary sources represent the results of more recent research.

5.16.4 Tips

What's Enough? Do not feel you must populate an index with hundreds of articles. Often, answers require only a few articles. Since searches return results sorted by relevance, it is often sufficient to only download 5 of the best citations to answer most common questions. For pathways start with 2 or 3 secondary citations for each of the primary sources. This saves time, money and minimizes rate limits. Build indexes incrementally over time.

Pathway References: When asking a question about pathways in particular, explicitly mention the pathway. For example, "How is smoking related to the KEGG NSCLC pathway?" is more likely to reference both the pathway for NSCLC (KEGG [hsa0522](#)) and the disease (MeSH [D002289](#)).

Narrowing Context: Since LmRaC also generates a compound query based on the whole question, make sure to include other context cues (i.e., not just genes and disease) that will help capture journal articles relevant to your particular experiment.

How Detailed? More detailed answers aren't always better. Since the requested complexity (i.e., detail) determines the number of sub-questions

generated, detail should be correlated with the complexity of the question, otherwise LmRaC will likely generate significantly redundant answers. Ask for more detail when there are expected implicit questions in the original question. Or, simply make those implicit questions explicit in your question.

5.17 Usage - Experiments

A key feature of LmRaC is its ability to answer questions about a user's own experiments and data. Two components make this possible:

- a user-defined experimental context of documents and data
- user-defined functions that answer questions about that data

The context is simply an index, like those created for general questions, that focuses on information about the experiment. Functions are user provided code that retrieves or otherwise manipulates data so that it is available to answer a question. A user needn't "call" the function explicitly, he or she need only describe it and then leave it to LmRaC to use the function if it will aid in answering the question.

5.17.1 Experimental Results

For example, assume you have an experiment where you have measured differentially expressed genes (DEG) between affected (disease) and unaffected (control) subjects. Typically, this type of experiment results in a file that contains all measured genes, how much they changed between experimental conditions, and the statistical significance of that change. You could ask:

```
[user] What is the expression of BRCA1 in my experiment?
```

[LmRaC] The expression of the gene BRCA1 in your experiment "greatStuff" is as follows:

- Log Fold Change (logFC): 1.488
- Adjusted p-value (adjP): 0.000001

LmRaC first notices that this is a question about your experiment. Internally it identifies the current experiment as "greatStuff." Then, in attempting to determine the expression of the gene BRCA1 it finds a loaded user-function with the description "Return gene expression results for a list of genes from an experiment." It creates a list with BRCA1 as the only gene and calls the function. The function reads the DEG results, finds BRCA1, and returns its expression. LmRaC incorporates this into the answer above.

See *Usage: User-Defined Functions* for how to create your own functions.

5.17.2 Creating an experimental context

When asking questions about experiments LmRaC will first search for documents indexed for the experiment. This means documents in the experiment's *docs/* folder. These are either saved answers or uploaded documents (see below). In either case these documents have been copied to the *docs/* folder and, most importantly, had their embeddings computed so that they are available for search.

Creating a context, therefore, means creating a group of documents that provide a focused knowledge base relative to the experiment. This can be background documents on genes, pathways and diseases generated from general questions. It can also include relevant protocol or other background documents that provide details relevant to interpreting your experimental results.

Importantly, one of the strengths of LmRaC is that though it will use this experimental context, it can also search the general literature. Keep this in mind when formulating your questions.

IMPORTANT Experiment documents are "authoritative" when asking experimental questions. LmRaC does not implicitly use GPT4 to validate or otherwise confirm the veracity of any document that is uploaded. Therefore, for example, if you upload experiment documents that provide evidence that the world is flat, expect answers to use this "fact." You are the author of your experimental findings!

Saving Answers to Experiments

From the *Answers Window* select any answers you want saved to a specific experiment. Select an index. Click on the copy documents icon. Keep in mind that when an answer is saved to an experiment, the text of the answer is saved in the experiment directory, but the embeddings (i.e., searchable meaning) are saved in an index along with metadata that links it to the experiment. When asking questions about the experiment, you must use this index.

Manual document upload

Documents can also be uploaded to experiments manually. Simply ask LmRaC to "load experiment document to <experiment>". You will be prompted for the document name. Once copied to the experiment's *docs/* folder, embeddings will be computed and added to the current index. This makes the document available for questions.

Although it is possible to simply copy documents into the experiment's *docs/* folder, embeddings will not be computed, therefore, the documents are not searchable when asking questions about the experiment. Use the "load experiment documents" command

to make the document available for search.

IMPORTANT Docker containers can only see directories that have been mounted using the `-v` command. This means the path to upload documents is relative to the container's mount point. For example, if you run the LmRaC container with `-v $(pwd)/work:/app/user`, LmRaC can only see folders in the `work/` directory tree. These are referenced in the container as `/app/user`. So, if you want to load a document from `work/my-docs/experimentInfo.txt`, the full path when using the LmRaC load command would be `/app/user/my-docs/experimentInfo.txt`.

5.17.3 Putting it all together

First, create an experimental context for the `doug1` experiment:

```
[user] Describe the expression patterns of BRCA1 in breast cancer
```

Save this answer to the `doug1` experiment *docs/* using the *Answers Window*. Index the answer using the `dougi1` index.

Now, ask how my particular DEG results might be interpreted with respect to this context:

```
[user] set experiment = doug1
```

```
...
```

```
[user] set index = dougi1
```

```
...
```

```
[user] What is the expression of BRCA1 in my experiment doug1?
```

My experiment is studying breast cancer.

How might this expression be interpreted with respect to what the `doug1` experimental documents say about BRCA1 and breast cancer?

LmRaC provides the following answer:

[LmRaC] The expression data for BRCA1 in your experiment "doug1" shows the following:

- **Log Fold Change (logFC)**: 1.488
- **Adjusted p-value (adjpv)**: 0.000001

Interpretation:

1. **Log Fold Change (logFC)**: A logFC of 1.488 indicates that BRCA1 is upregulated in your breast cancer samples compared to the control. This means that the expression level of BRCA1 is higher in the breast cancer tissues.
2. **Adjusted p-value (adjpv)**: The very low adjusted p-value (0.000001) suggests that this upregulation is statistically significant, meaning it is highly unlikely to be due to random chance.

Contextual Interpretation:

BRCA1 (Breast Cancer 1) is a well-known tumor suppressor gene. Mutations in BRCA1 are associated with an increased risk of breast and ovarian cancers. However, the upregulation of BRCA1 in your experiment could indicate a cellular response to DNA damage or stress, as BRCA1 is involved in DNA repair mechanisms.

In the context of breast cancer, the upregulation of BRCA1 might be a compensatory mechanism where the cells are trying to repair DNA damage more actively. Alternatively, it could reflect a specific subtype of breast cancer where BRCA1

expression is higher.

To fully understand the implications, you should refer to the specific experimental documents of "doug1" to see how BRCA1 expression is discussed in relation to breast cancer. The documents might provide insights into whether this upregulation is consistent with previous findings or if it suggests a novel aspect of BRCA1's role in breast cancer in your specific experimental conditions.

5.18 Usage - User-Defined Functions

LmRaC functionality is designed to be easily extensible. That is, the range of questions that LmRaC can answer is only constrained by the imagination (and industriousness) of the user. Though primarily intended to facilitate access to user data and results, user-defined functions can also be used to interface to any other resource whether static (e.g., database, resource API) or dynamic (e.g., instrumentation, sensing).

A functionally complete example of the REST API is provided in this GitHub repository in the *RESTserver* folder. How to build and extend it is described below.

5.18.1 Quick Start

If you want to try the REST server without building it, a Docker image is available. Keep in mind that this example has limited functionality, but it will give you a quick way to try functions.

NOTE this Docker image has been built from the provided GitHub example (*RESTserver*).

Pull the latest tagged image from Docker Hub. Run LmRaC REST using Docker Engine.

If Docker is not installed or you're using Docker Desktop, see the *Installation* instructions above. This example server does not use any API keys.

IMPORTANT you must mount the same local directory as was used for LmRaC otherwise the REST API will be looking in a directory different from LmRaC and will not find the data files it is returning results from. Also, notice that the REST server is on a different port than LmRaC (typically 5001).

```
docker pull dbcraig/lmracrest:latest
cd <your-lmrac-root>
docker run -it -v $(pwd)/work:/app/user -p 5001:5001 dbcraig/lmracrest
```

On startup the LmRaC REST server will print the IP:port it is running on. Make sure that this matches the LmRaC configuration otherwise LmRaC requests will not be received by the REST server. To aid in debugging, this example server echos requests it has received. If you do not see a request, then they are likely being sent to the wrong IP:port address.

The exampleExp Experiment

Included with LmRaC configuration is an example experiment: *exampleExp*. This experiment includes sample data files that are compatible with the *DEGbasic* function library (see below). Set *exampleExp* as the current experiment. Load the *DEGbasic* functions. You should now be able to ask questions about experimental data (e.g., "What is the expression of BRCA1 in my experiment?") as part of your questions to LmRaC.

5.18.2 Building your own REST API Server

You can use the provided example as a starting point for building your own REST API server. Clone the base REST API server from this GitHub repository, then build the Docker

image for the functions server.

```
git clone https://github.com/dbcraig/LmRaC.git
docker build -t lmracrest:latest .
```

Run the functions REST API server:

```
cd <your-lmrac-root>
docker run -it -v $(pwd)/work:/app/user -p 5001:5001 lmracrest
```

When the server starts up it will show the IP:port on which it is running.

LmRaC must know this IP:port in order to make API requests. You can edit the LmRaC.config file (see *Configuration*) so that the *functionsREST* key value is set to IP:port (e.g., "172.17.0.2:5001") or you can set the IP:port dynamically by asking LmRaC to set the value (e.g., "Please set the functions REST API IP and port to 172.17.0.2:5001")

5.18.3 Adding Functions (Server - REST Server)

The REST API example includes the following files and folders:

- **RESTserver.py** Initializes all functions and starts the server.
- **DEGbasic.py** An example library of functions for returning results from differential gene expression experiments.
- **Dockerfile** The script for building the Docker image.
- **requirements.txt** A list of additional Python libraries to include in the Docker image.
- **.dockerignore** Files and folders to ignore as part of the Docker build process.
- **data/KEGGhsaPathwayGenes.tsv** DEGbasic function specific data to associate genes with pathways.

Follow these steps to add functions to the REST API server.

RESTserver.py

1. Import your functions.

```
from DEGbasic import *
```

2. Instantiate your functions. This also performs any initialization by calling the `__init__` method.

```
DEGbasic = DEGbasic()
```

3. Add your function names to the API. This allows LmRaC to make a request by name.

```
functions.update( DEGbasic.functionsList() )
```

The `functionsList()` is a require method for all user-defined function classes.

DEGbasic.py (or your custom functions file)

4. Import needed libraries. Add these to the `requirements.txt` file if necessary.
5. Define the `__init__` method to perform any needed initialization for your functions.
6. Define the `functionsList(self)` method to return a dictionary of named functions.

```
def functionsList(self):
    """
    Functions are called by name in the REST interface.

    This maps the function names to the actual functions.
    """

    return {
        'initializeDEGbasic' : self.initializeDEGbasic,
        ...
    }
```

}

7. Define your functions.

- **Naming:** Function names should be the same as in the LmRaC functions prototype file (see *function prototypes* below). Function names *are* case sensitive. Note: do not include the prefix *lmrac_* on function names. This is added internally during compilation.
- **Parameters:** Parameters are passed in as JSON. Use *json.loads()* to parse the parameters into a Python object. Be sure to handle optional parameters and any defaults they may have.

```
params = json.loads(params)

top_k = params['topK']

experiment = params['experiment']

if ( 'filename' in params ):

    filename = params['filename']

else:

    filename = 'pathwaySig.csv'
```

- **Return values and errors:** All functions must return a string value. This may be free text, structured JSON, or some combination.

IMPORTANT If your function needs to return an error, make the error descriptive and helpful to answering any question. To force LmRaC to abandon answering a question due to an error, prefix the return text with **Function Error** followed by the function's name prefixed with **lmrac_**, as follows:

```
return "Function Error lmrac_<function-name> ..."
```

5.18.4 Using Functions (Client - LmRaC)

To make functions available to LmRaC simply create a function prototype file in the *functions/* folder. The file must end with a *.fn* extension. When LmRaC starts it will read all *.fn* files, compile those that do not have a current *.json* file, and then make those available for loading (recall that functions must be *loaded* once LmRaC has started to be available to answer questions).

IMPORTANT Prototype file names *are* case sensitive.

Function prototypes

LmRaC, the client, needs to know when and how to call functions. This is accomplished by providing function prototypes. The prototype includes the following:

- function name
- function description
- function parameters

LmRaC calls functions based on the function description. This is important: descriptions are not comments, they are integral to choosing functions to call. Parameters are passed using information either from the original question, or from previous function calls. All parameters are typed and must include a description. Descriptions are used by LmRaC to choose the correct value to assign to the parameter. Parameters may be optional. Allowed types are:

- NUMBER
- STRING
- BOOLEAN
- ARRAY

All functions must return a string. This reply may be free text (natural language), structured text (JSON), or some combination. This text is returned to LmRaC as supplemental information to answer the original question. Think of it as additional information you *could* have added to your original question.

A simple prototype file looks like this:

```
#  
#      Line comments begin with a '#' mark  
#  
DESCRIPTION      "This describes the entire group of functions and is used to create  
a README file in the functions/ folder"  
  
FUNCTION initializeMyFunctions  "Functions dont have to have any parameters"  
  
# indentation is not necessary and is only used for readability; extra tabs and  
spaces are ignored  
# all lines must have a "description" value since these are used by LmRaC to  
interpret and assign values
```

```

FUNCTION getTopExpressionResults "This text describes what the function does and is
how LmRaC decides to use it"

PARAMETER topK:NUMBER          "The NUMBER type may be integer or float depending
on the function implementation"

PARAMETER byFoldChange:BOOLEAN "The BOOLEAN type is true or false"

PARAMETER experiment:STRING   "STRING type are for characters"

PARAMETER description:STRING* "Adding a '*' makes the parameter optional"

PARAMETER geneArray:ARRAY     "ARRAY types can contain any type item except
another array"

ITEM gene:STRING             "array items begin with the ITEM keyword"

```

Save all function prototype files in the *functions/* folder. The file name will be the name used when loading. Names are case sensitive.

Function prototype compilation

When LmRaC initializes it will attempt to compile all *.fn* files in the mounted *functions/* folder, unless the *.json* file (compiled version) is newer than the source (*.fn*). Upon successful compilation (i.e., no errors), the *.json* file will be available to LmRaC for loading.

Any error in compilation will be reported during LmRaC initialization. This is not fatal, but makes the functions unavailable for loading. Fix any errors and restart LmRaC. Errors are also detected when functions are loaded.

Function loading

Once LmRaC has started you may *load* functions to make them available to answer questions. Simply ask LmRaC to load the functions or use the *Functions Window*. The name of the functions library is the name of the prototypes file.

IMPORTANT Prototype file names *are* case sensitive.

Remember that LmRaC chooses functions based on their description. For example, if you ask a question about gene expression in your experiment, LmRaC will look for any functions that describe themselves as "getting gene expression." Describe your functions as accurately and succinctly as possible.

When loading functions LmRaC does a final quick error check. If the `<function>.json` has errors these will display as an error message when loading from the LmRaC dialog or as a red exclamation icon in the *Functions Window*. Hovering over the error icon will show the error (Figure 28).



Figure 28: LmRaC error during function load.

Function unloading

Only load functions when they are needed. Since functions are passed to GPT4 whenever a question is asked, they are included as part of the input token count. This means that aside from being unnecessary (and possibly confusing) when asking a question, they also incur unnecessary API costs.

5.18.5 General Functions

Though the functions described here read and manipulate data, LmRaC functions can also be used to make other information available to LmRaC (e.g., retrieving information from APIs or search). Try it! Let us know what useful functions you've defined and how you've extended LmRaC's functionality.

5.19 Indexes and Experiments and Functions

Indexes are purposely independent of Experiments.

Functions are purposely independent of Experiments.

With flexibility comes responsibility. Maybe the most confusing part of LmRaC is remembering that indexes and experiments are independent. Think of indexes as a collection of information about a domain of knowledge *that you define*. Within that collection is additional information about particular experiments (e.g., answers you've saved, documents you've uploaded, interpretations of results). So, if you want to ask questions about (i.e., search) these experiments, you need to use the correct index.

However, information about an experiment can be saved to *more than one index!* You may have an index created specifically for one disease (e.g., breast cancer). You may have another index created for a particular experimental protocol (e.g., differential gene expression). Information about your experiment investigating gene expression in breast cancer can be saved to both indexes! *Or*, you could just create one large index for both breast cancer and differential gene expression. You have the flexibility to design the knowledge base best suited for the questions you ask.

Likewise, functions are designed to manipulate your data (e.g., read, search, compute). You can group them any way you like and use them in any combination. They are not part of an experiment. They are only there to aid in answering questions about the experiment.

5.20 Troubleshooting

LmRaC isn't calling my function: The most common cause for this is that the function has not been loaded. Although function files are read and compiled at initialization, they must also be *loaded* in order to be available for

questions. This allows LmRaC to focus only on functions relevant to the task at hand. Note that which functions are loaded is saved to the configuration file, so after restarting LmRaC your functions are automatically re-loaded.

LmRaC won't stop calling my function: If your function does not return what it's description promises it should, LmRaC will try again. And again. And again. Make sure to perform the described function. If there is an error, return **Function Error** followed by the function's name prefixed with **lmrac_**. This will signal to LmRaC that the function has failed and not to try again.

Memory: Because LmRaC uses multiprocessing extensively, complex questions can require significant memory resources while documents are being processed. We recommend a minimum of 1GB for the Docker container, though 2GB may be necessary for large multi-part questions. The error **A process in the process pool was terminated abruptly while the future was running or pending** is usually an indication that LmRaC ran out of memory.

Rate Limits: All servers have rate limits (i.e., maximum number of requests per second and/or per day). In the case of PubMed this is fixed. For OpenAI this increases over time for users. In all cases LmRaC will retry a request in the event of a rate limit error. Retries employ an exponential backoff strategy that, in most cases, is sufficient for the request to ultimately succeed. As a consequence, users may see slower response times when using LmRaC with a new OpenAI account. Keep this in mind when first building an index since computing embeddings is also a rate limited activity.

Low Assessment Scores: Note that it is not unusual for GPT4 to assess final answers as poor. Most often this is due to two factors: (1) GPT4 flags citations as "fake" because they occur after the training cutoff date of GPT4; or, (2) GPT4 objects to the complexity of the answer as exceeding the scope of the original question, or inappropriate for a lay audience. On the other hand, these assessment often offer insightful critiques that may prompt further questions.

5.21 How To Cite

Douglas B Craig, Sorin Drăghici, LmRaC: a functionally extensible tool for LLM interrogation of user experimental results, *Bioinformatics*, Volume 40, Issue 12, December 2024, btae679, <https://doi.org/10.1093/bioinformatics/btae679>

5.22 Contact

Douglas Craig : <craigdou@med.umich.edu>

CHAPTER 6 LMRAC COLLABORATIVE NOTES

6.1 Abstract

Human-centered artificial intelligence (AI) systems are most effective when they foster collaboration with users, enabling iterative problem solving and creativity. However, many existing AI solutions operate as opaque, autonomous systems, limiting opportunities for human engagement, transparency, and refinement. This lack of integration between AI and human expertise often results in barriers to trust and hinders the development of innovative solutions, particularly in complex, data-intensive domains like scientific research.

Here we introduce a novel human-LLM system built on the Obsidian note-taking application, designed to integrate large language models (LLMs) into a transparent, interactive, and collaborative framework. Inspired by the Zettelkasten [192] method, key features of this system include case-based reasoning (CBR) and language-aware tools, with cases and tools represented as first-class notes. A Python executive program coordinates user interactions, while advanced search capabilities powered by embeddings and a graph database enhance the retrieval of relevant cases and tools. In addition, the system supports techniques and tools for incrementally building and refining solutions to novel problems, thereby facilitating both structured workflows and scientific discovery.

The system tackles the challenges of integrating AI into human workflows by promoting transparency, adaptability, and meaningful collaboration. Initially tailored for the biological sciences domain, it enhances productivity and insight by enabling a seamless, interactive partnership between humans and AI, advancing the state-of-the-art

in scientific workflows through a unified and accessible platform.

6.1.1 Keywords

collaborative software, large-language models, case-based reasoning, knowledge discovery, scientific workflow

6.2 Introduction

Scientific workflows are complex processes often involving iterative problem-solving, collaboration, and the integration of diverse tools and datasets. Advances in artificial intelligence (AI), particularly large language models (LLMs), offer significant potential to enhance these workflows by providing intelligent, context-aware support. However, many existing AI systems function as autonomous black boxes, providing solutions without transparency or opportunities for user engagement. This lack of interaction limits their ability to foster creativity, trust, and iterative refinement—critical components of effective scientific inquiry.

This work introduces a novel human-LLM note-taking system designed to address these limitations by embedding LLM capabilities within a human-centered, note-centric framework. Built on the Obsidian note-taking platform [158], the system integrates case-based reasoning (CBR) [193, 194] with LLM-enabled tools, allowing users to collaboratively solve problems while maintaining transparency and traceability. Unlike typical question-and-answer systems, our approach treats users as active participants in the workflow, allowing them to inspect, revise, and adapt the system’s processes and solutions.

The system introduces several innovative design elements to enhance functionality and

usability. Cases and tools are represented as first-class notes in Obsidian, making all system components accessible for inspection and revision. A Python executive program manages interactions between the user, LLM, and note corpus, facilitating seamless integration. The use of embeddings and a graph database enables sophisticated searches for relevant cases and tools, while bootstrapping and progressive problem-solving techniques enable users to incrementally develop solutions to novel problems.

Through this collaborative framework, the system seeks not only to improve the efficiency of scientific workflows, but also to foster serendipitous discovery and creative problem solving. By integrating human expertise and machine intelligence within a shared workspace, this system provides a transparent and consistent environment for advancing scientific inquiry. Currently focused on the biological sciences domain, the system supports dynamic creation of retrieval-augmented generation (RAG) [48] document collections, allowing GPT4o [21] to be supplemented with authoritative sources. This paper describes the system’s design, operation, and early results, demonstrating its potential to transform the way humans and AI work together.

6.3 Methodology & Design

Keeping with a human-centered approach, the system is designed to be note-centric. That is, all major components of the system are first-class text documents (i.e., markdown notes) in the Obsidian note-taking application. This includes not only user notes but all system CBR cases as well as tool specifications. This approach means that all steps of the workflow are transparently available to both the user and LLM for inspection and revision. The only exception to this is a system console that facilitates a meta-dialog for actions and

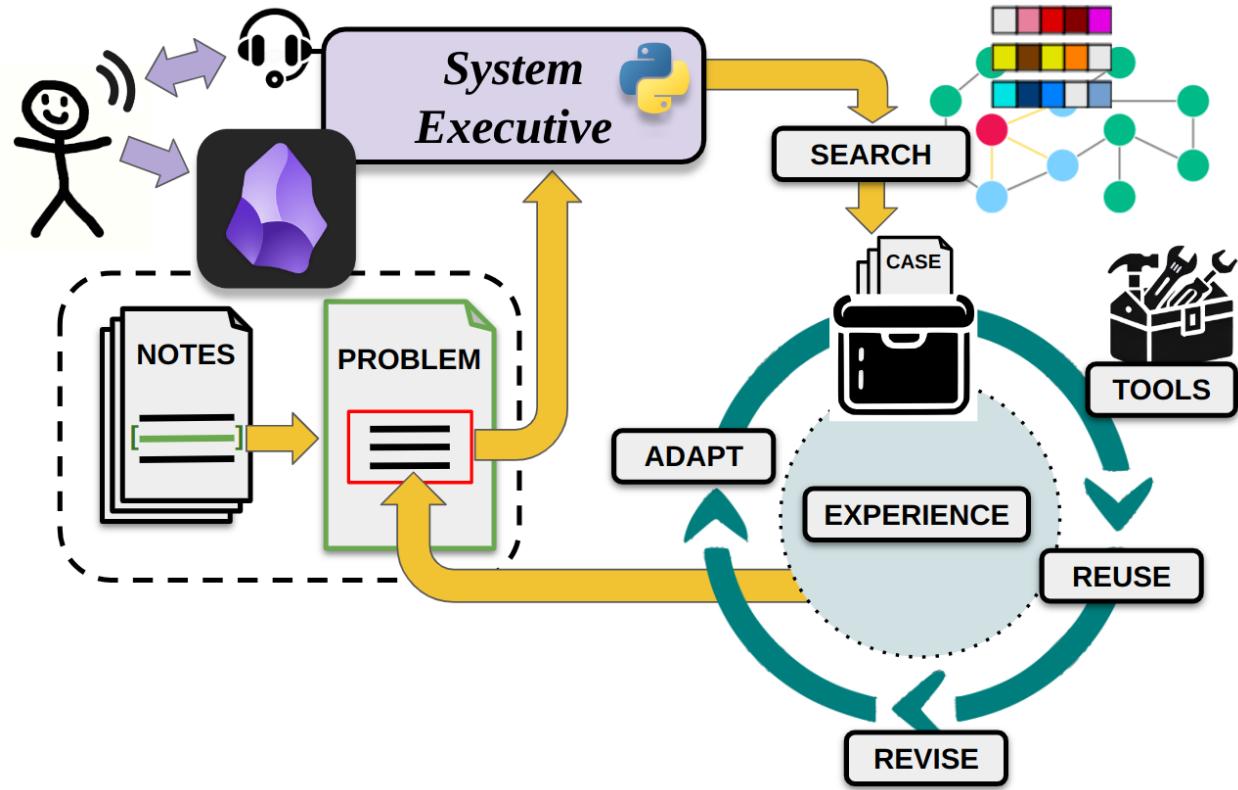


Figure 29: System workflow. The user interacts with notes and system console. Notes may identify a problem for which the System Executive will search for previous case instances. Cases are themselves notes that utilize tools to access external functions and data. A case is then reused or adapted to the current problem.

questions on or about notes or processes.

6.3.1 System Workflow

Figure 29 gives an overview of the overall workflow. The user interacts directly with the note-taking application (Obsidian) or with the System Executive through voice or keyboard. While taking notes, the user may prompt the system to evaluate a change to the current note. The system evaluates the change (relative to a previously cached version) to assess if there is a question or problem to be addressed. If so, it searches for an applicable case. The search uses a combination of similarity (embeddings) and semantics (graph database). A new instance of the most similar case is then created and linked to from the

current note.

6.3.2 Case-based Reasoning

Case notes are structured documents that encapsulate knowledge for solving problems. Each case includes: a description of the problem, background information and/or resources, and steps for solving the problem. The case typically also includes references to tools necessary to complete the steps. Tools may be specified explicitly by name or, more generally, by description. The latter case is used when tool selection depends on the circumstances of the case's application. Cases represent the system's experience (i.e., memory) of previous problem solving instances. As such, cases may be simply reused or, more commonly, revised to improve performance or adapted for new problems.

6.3.3 The Collaboration Process

When a note links to a case, the executive begins execution of the steps necessary to solve the problem or answer the question. Because the case is a note, the user sees not only what steps are to be executed, but the results of those steps. Through the console, the user may pause execution to review, revise, and/or repeat steps. This makes the process interactive, transparent and traceable. The system also supports ignoring any portion of the results as well as entire steps using markdown strike-through. This facilitates documentation of previous attempts/failures. Since all note text is available to tools (e.g., GPT), the system is always aware of the state of progress as well as any user feedback.

Bootstrapping

As the system is exposed to new problems and questions it is necessary to create new cases. This is handled through a GPT enabled bootstrapping tool (Figure 30). When a

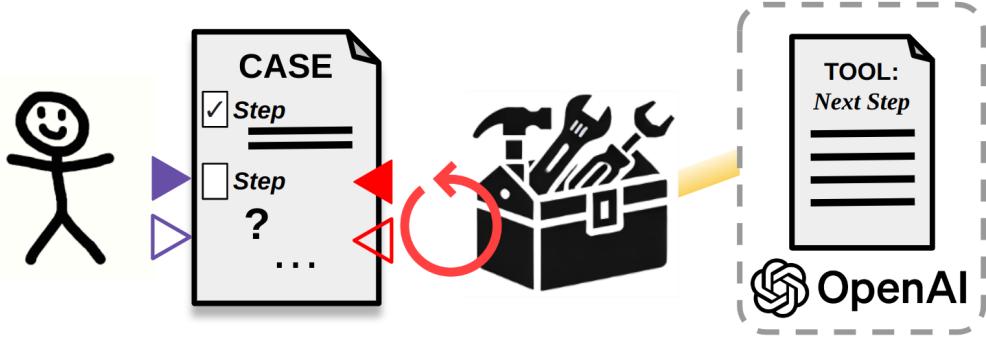


Figure 30: Bootstrapping and progressive problem solving. A case may initially only include a starting point. From this the user, in collaboration with the LLM, can incrementally build a solution to a problem. Cases may also use this approach to progressively solve problems with user input/feedback.

novel problem is detected (i.e., one for which the case "I've never seen a problem like this before" is the best match) the first step is to use the *Next Step* tool to prompt GPT to generate a starting step for solving the problem. This may result in a case step to: gather information, ask for user guidance/suggestions or search for a tool appropriate for the solution. In the case that a suitable tool is not found, the *Suggest Tool* tool can describe the tool it needs to the user.

Progressive Problem Solving

Similar to how a user may pause step execution at any point, it is also possible to use *Next Step* as an integral part of any case. That is, for cases where the manner in which a problem is solved are conditional yet constrained, the tool can "build" the case as each step progresses. In this case *Next Step*'s ability to use the context of prior steps is leveraged to provide flexibility in the solution of a class of problems.

6.3.4 Language Enabled Tools

As mentioned, tools themselves are notes. That is, each tool includes a note specifying its use and application. This allows the note to be searched, like cases, based on similarity

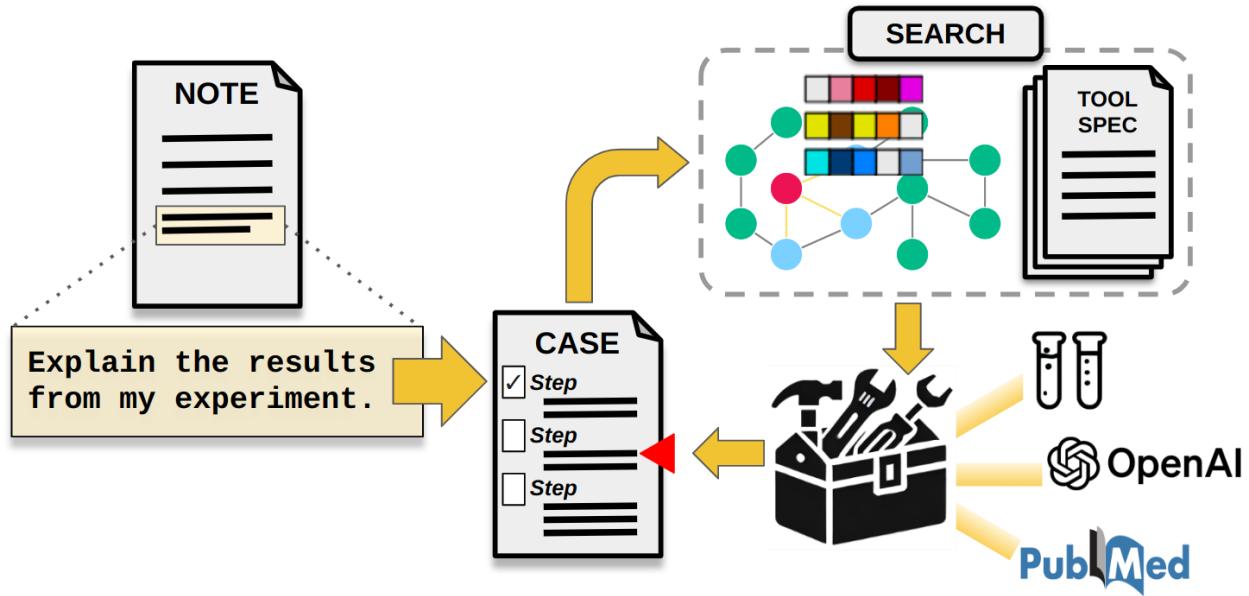


Figure 31: Tools selection. In posing a question a case is retrieved. This case utilizes tools either explicitly, or by search based on a description of the problem and data for which a tool is suited. Tools are applied stepwise to effect a solution for a given case.

and semantics (description). Function notes also specify input parameters and output. For most tools this is formalized as in any functional programming language so that back-end code can implement the function. However, functions can also perform language functions (e.g., summarize). For example, in Figure 31 the note asks for an explanation about a user's data. The associated case relies on tools to: retrieve local data files, analyze the data with respect to the larger note context (e.g., a differential gene experiment), and compare these to the literature. Note that, depending on context and user preference, this explanation may be fast and informal (based on GPT) or thorough and formal (with citations to authoritative sources). In the latter case, tools are available to build and access user constructed domain knowledge (RAG). Tools that retrieve or manipulate data may take advantage of an extensible user-defined REST API (e.g., to retrieve data from third-party databases).

6.3.5 Implementation

The system is implemented using Obsidian with a Python executive. OpenAI's GPT4o model is used for LLM-based tools. A Pinecone (<https://www.pinecone.io/>) serverless vector database maintains embedding for both search and custom RAGs. neo4j (<https://neo4j.com/>) is the graph database used to capture semantic relationships for cases and functions.

6.3.6 Executive Execution and Interface

The primary interface to the system is through the standard Obsidian note editor. Notes are plain text with markdown. This choice for Obsidian as well as LmRaC is that it makes notes "future proof," that is, notes *are not* stored in a binary or otherwise proprietary format. It also ensures that all notes are natively readable by both human and machine.

Front-matter

Note metadata is likewise stored as plain text at the start of every note. To make this machine readable it is formatted as YAML, a standard used primarily for configuration and data control (Figure 32). The LmRaC executive uses the *inotifywait* utility to efficiently monitor all Obsidian files for changes. Changes to the front matter signal to LmRaC that the user is seeking attention by setting the ENQ flag.

Compute Note Difference

A changed note is then compared to the most recently cached version to determine *what* has changed at which point LmRaC will initiate a case to determine the appropriate action (e.g., answer a question). This step uses the *Obsidian:saveFileToCache()* method. If a cached version of the file exists, then it computes the *diff* between new and old version

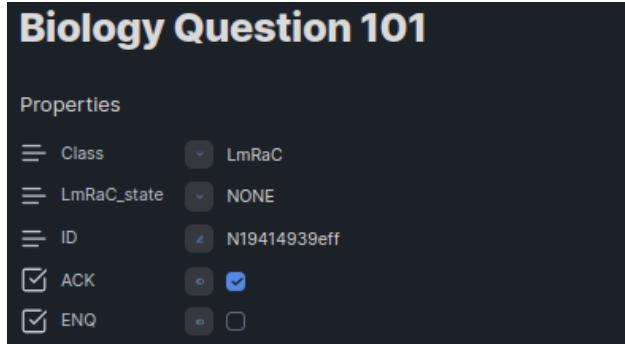


Figure 32: All LmRaC notes include front matter (stored as YAML in the markdown file). This metadata identifies the note as a LmRaC note, its unique identifier and current ENQ/ACK state.

and then the cache is updated. If no cache exists then the *diff* is simply the whole file. An asynchronous message is sent to the dispatcher with the note difference as message payload.

Process Note Difference

The *Obsidian:processNoteDiff()* method converts output from the Unix *diff* (i.e., hunks) into a list of added/changed lines. Warnings are issued for: no changes found (e.g., only white space) or if there are multiple changes too far apart in the file. GPT is then used to check if the changes are a biological question or problem. If so, a case search is initiated.

Figure 33 shows feedback on the LmRaC console for this step.

Case Instantiation

The chosen case is then copied (*Obsidian:createNote()*) and a link is created from the user note to the new version of the case (see Figure 36 for how this looks in the note, along with the backlink to the original case). This instantiated case will form the starting basis for answering the question or solving the problem. Keep in mind that each instance of a case is a *proposed* solution. It is not fixed. The user may suggest or make changes. LmRaC may revise the solution steps. Both are in the service of adapting to new and

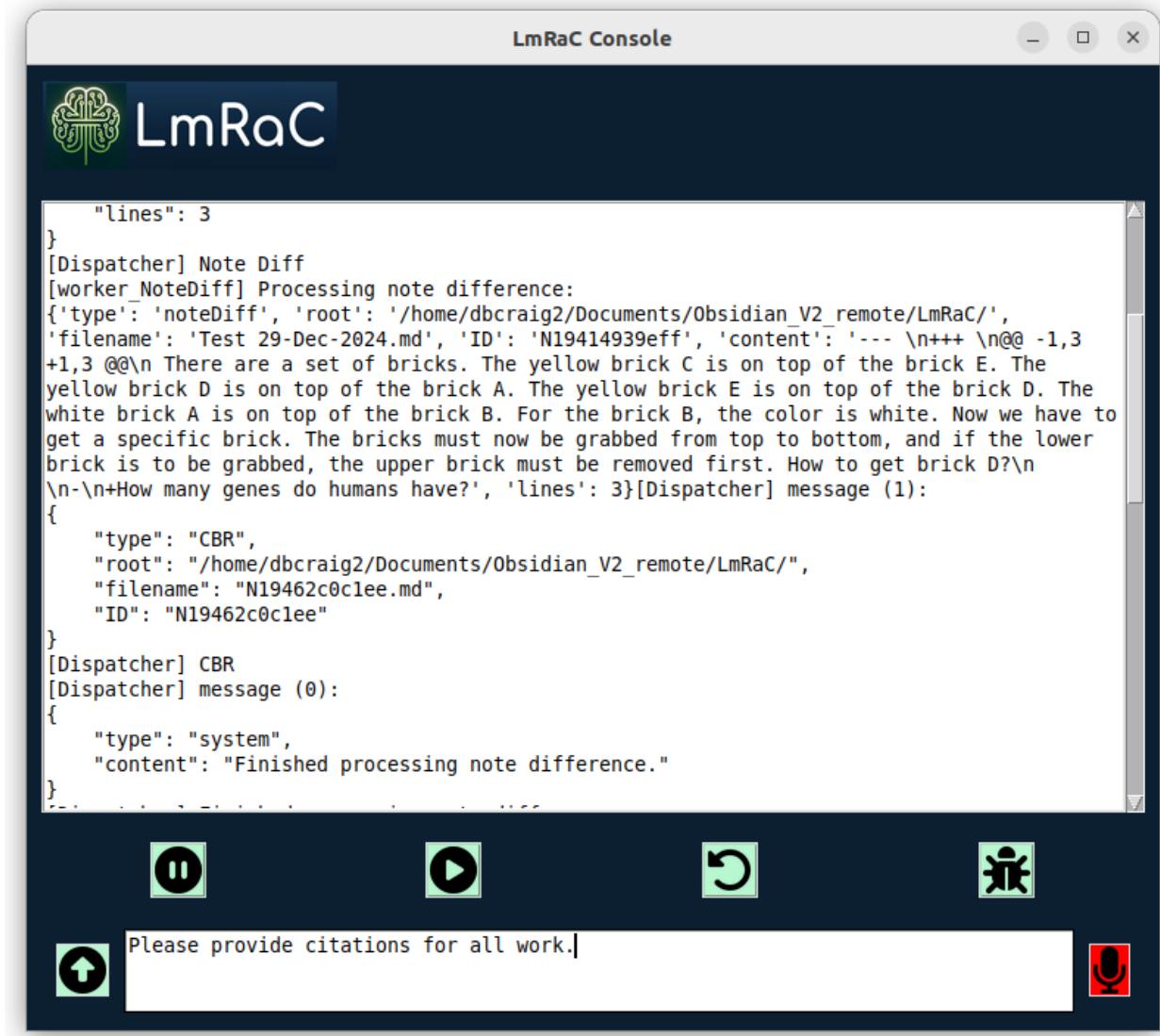


Figure 33: The LmRaC console application provides the main user interface outside of Obsidian. It provides debug feedback as well as meta-control buttons for: pause, resume, retry and debug as well as user text and voice input.

novel questions. Once created and linked a message is pushed to initiate interpretation (*LmRaC:messageStack_push()*) and execution of the case (*LmRaC_notes: dispatcher()*).

Case Execution

Cases are processed one step at a time in a cycle that allows for return information/values to be incorporated as execution progresses. This is accomplished through plain text or JSON that is added to the case document. Any JSON is then added to the case context making it available for reference/substitution in the case. This process allows for an iterative process of building a solution to the case. It also allows the user to inspect intermediate methods and results. At any point the user may pause execution to inspect or intervene.

Case Notes

As mentioned, a key design strength of LmRaC is that cases and tools are first-class notes, just like user notes. This allows the user to act as an agent in the reasoning process. Case notes are stored in the *cases/* folder (Figure 34). Cases include a Description of the problem the case addresses. When the case note is created, and any time it changes, a vector embedding is created for this description and stored in the database. This provides the mechanism for cases to be found via a simple search that compares any question to available cases.

Tool Notes

Likewise, tools notes are stored in the *tools/* folder (Figure 35). Tools include both a Description as well as a declaration for their interface. Like case notes they are indexed using embeddings and searchable.

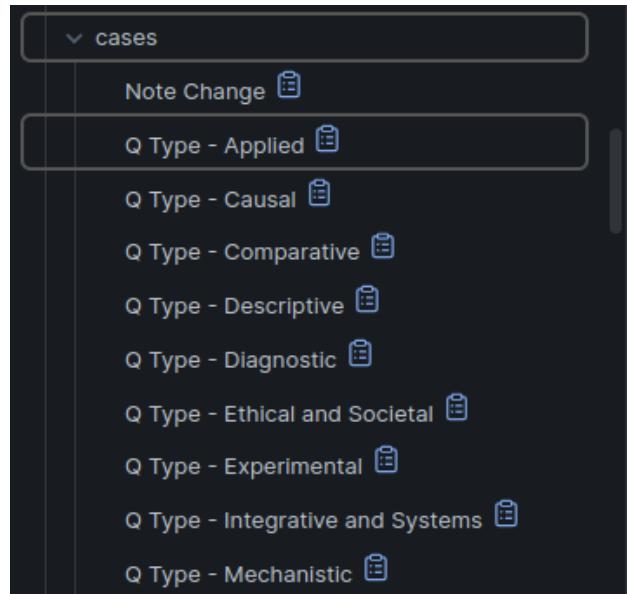


Figure 34: CBR cases are stored as Obsidian notes.

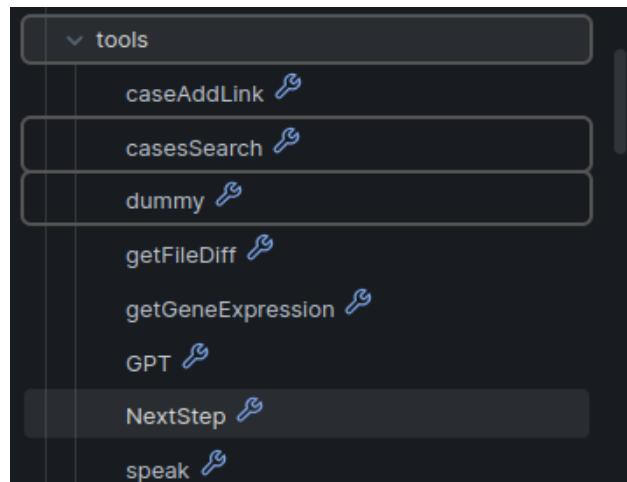


Figure 35: Tool specifications are stored as Obsidian notes.

Case Execution

Figure 36 shows a simple question – How many genes do humans have? – asked in the context of the note *Biology Question 101*. The case Note Change is instantiated to evaluate the change when the question is asked. The tool *getFileDiff()* is used to effect the file difference and return the changed text as a embedded variable, *diffText*. LmRaC automatically substitutes this return value into the next tool which assesses if the change is a biological question. This makes all steps of the process transparent to the user who posed the question.

6.4 Results

6.4.1 System Functionality and Usability

This system demonstrates productive integration of AI capabilities within an established note taking workflow. The Python executive program operates seamlessly with Obsidian, monitoring note changes in real-time and triggering appropriate analyses without disrupting user workflows. Conversely, the user can at any time interrupt or prompt the executive to ask for explanation, offer suggestions or initiate solution of a problem. This allows human and machine to work in concert on a common task. By exposing all case reasoning and tools through a common notes interface the language strengths of LLMs can be leveraged while still maintaining an intuitive and transparent interface.

6.4.2 Innovations and Contributions

The system’s design introduces several innovative elements that enhance its utility and applicability. Although case-based reasoning (CBR) has a long history going back

Biology Question 101

Properties

Class	LmRaC
LmRaC_state	NONE
ID	N19414939eff
ACK	<input checked="" type="checkbox"/>
ENQ	<input type="checkbox"/>
+ Add property	

How many genes do humans have?

N19462c0c1ee



Case Template: [LmRaC/cases/Note Change](#)

Backlink: LmRaC/Test 29-Dec-2024.md

Problem

When a note changes, see what changed and decide if this is a biological problem. If so, identify the general problem type.

Solution

Step : Assessment

Step Completed

Requires

```
diffText = tool:getFileDiff()
>{"diffText": "How many genes do humans have?"}
```

Action

```
{"functions":"
FUNCTION assessBiologyQuestion 'Assess text based on if it is a biological question'
PARAMETER isBioQuestion:BOOLEAN 'True if the text is a biological science question or problem. False otherwise.'
',"prompt":"
Assess the following text to determine if it is a question or problem related to biological science.
Reply with True or False.
Do not solve the problem or otherwise process the text.
```

Figure 36: When a question is asked within a note the *Note Change* case is used to assess the type of question. A new instance of *Note Change* is created and referenced from within the user note using internal link markdown.

to the early days of AI [195–197], integrating LLMs with CBR is still nascent [194, 198, 199]. It is well established that cases provide a natural mechanism to capture and build upon experience [200] and provide a natural extension for making AI explainable [201]. Additionally, case notes serve as both documentation and a foundation for iterative learning through a cycle of retrieve-reuse-revise-retain, making problem-solving integral to the workflow. The integration of a graph database with vector embeddings (aka GraphRAG), while not novel [202, 203], offers an efficient means of identifying optimal solutions (cases) and implementing them effectively (tools) within a text-based environment (notes). By wrapping all major components, including tools and cases, within Obsidian, the system delivers a transparent, cohesive, and consistent experience that encourages human-machine collaboration and enhances usability while leveraging the strengths of an LLM.

In contrast to many current autonomous systems that operate as proprietary question-and-answer black boxes, our system fosters progressive and collaborative problem formulation and solution development. By presenting both case formulations and results in a unified note format, the system naturally supports tools for reflection and review, enabling a seamless process of iterative refinement and user interaction.

6.4.3 Qualitative Results

Since the system is by design not an autonomous problem solver, it makes comparison to systems that are (e.g., OpenAI o1) difficult. In fact, the goal is arguably different: our work seeks to amplify current human scientific workflow by leveraging LLMs, CBR and other tools. Collaboration is the point. Having said that, preliminary results anecdotally demonstrate that integrating an LLM into a scientific workflow significantly enhances

insight and productivity. The system’s ability to suggest not just answers but approaches to finding answers within the context of a standard workflow is, we feel, the main result of this work. Beyond simply searching for an existing solution, the use of bootstrapping and progressive solution demonstrate how LLMs, in collaboration with humans, can discover *new* solutions.

6.5 Discussion

Although recent models like OpenAI’s GPT4o and o1 are adept at generating stepwise solutions to problems, they lack support for iterative refinement or learning from past mistakes. In contrast, our system maintains the human as equal partner, fostering a collaborative approach to problem-solving. By embedding cases and tools as first-class notes within a unified platform, we believe the system provides a transparent and consistent environment that advances scientific workflows. Moving forward, our work is focused on enhancing tools to facilitate adaptation and learning from past cases, and developing mechanisms for discovery and synthesis across notes. To achieve the first, we plan to employ multi-faceted similarity measures that take into account not only the overall objectives of a case but also individual steps and dependencies, enabling flexible and fractional repurposing of cases. To support discovery across notes we are developing tools that also search user notes. The challenge is in balancing constrained search while allowing for serendipitous association. To this end we are exploring methods that perform random walks in semantic graphs between notes. The idea is that by *not* taking the shortest path other more distant semantic associations can be discovered.

6.6 Future Work

In addition to enhanced similarity and cross-note discovery additional extensions and enhancements to tools and cases are planned.

6.6.1 Cases

Backlinks

Obsidian keeps track of all links to a note. Given a note it then allows the user to see what notes refer to the note, the backlinks. Currently, case instances include this backlink explicitly. This will enable easy collection and navigation of case notes and their progeny. This link provides the path by which LmRaC can potentially learn from previous experience by not just reusing, but revising and adapting previous case application for new problems.

Speech

LmRaC currently uses speech primarily to communicate to the user, but it also has integral speech-to-text. This is currently used the same way as typed text, but has the potential to create a more fluid interface whereby LmRaC and the user can converse while execution proceeds. Prototyping is underway to experiment with the OpenAI realtime voice mode. This would enable the application to keep a port open at all times and react more smoothly with user speech.

Initiation Cases

Although experiments have been done with the *Next Step* case, more work needs to be done to provide a general mechanism to initiating solutions. In many cases a question or problem cannot be matched to a productive case. It would be helpful to have a case that broadly applies a heuristic to improve framing of a problem and initiating further

exploration. This will be particularly helpful in discovery tasks where there simply is no established (or even hypothesized) formula for attacking a problem.

Case Creation

One purpose of maintaining cases as first-class notes in Obsidian was to open up the possibility of LmRaC itself to create de novo cases. Some work has been done to show that GPT-4o has a non-trivial ability to propose a step-wise plan for solving a problem. Using this as a starting point and then leaning on LmRaC's CBR framework to refine this plan as a draft case seems promising. Because cases are structured markdown documents the starting prompt becomes: "You are a reasoning system that relies on CBR to choose the best way to solve problems..." along with the JSON specification for a case.

6.6.2 Tools

Note Tools

Planned tools, which become available not only to the user directly through the console, but to any case, include:

- *Decomposition / Split* - break a note into parts (e.g., sentences, paragraphs, words) potentially using some more abstract operator that regroups/reorganizes
- *Merge* - combine notes, with or without smoothing (i.e., copy editing)
- *Lambda Function* - essentially, a one-off prompt acting as a function over a document
- *Semantic Analysis* - abstraction of notes for discovery (e.g., context, assumptions, argument form)
- *Structure-ize* - convert note to structured data for machine analysis (e.g., JSON)

- *Summarize / Reduce* - convert note to a summary or bullet list
- *Cluster* - define "distance"/likeness and cluster a group of notes (embedding calculation)
- *Classification* - group a set of notes both supervised and unsupervised
- *Topic Discovery* - using LDA [204] or other methods to discover topics in a group of notes
- *Labeling / Linking* - association of metadata ; automated; controlled vocabulary based; generate document hyperlinks to associate notes (supervised and unsupervised)

External Functions (APIs)

- *Multi-model LLM* - choose the likely most effective (by function and/or cost) model for a given case and/or context
- *Search* - generalized but, again, adaptive based on the given case and context; especially for authoritative documents
 - [Google Scholar](#)
 - [Wikipedia](#)
 - ChatGPT Search [205]
 - [PubMed](#)
 - [bioRxiv](#)
 - [medRxiv](#)

- *Genomic Databases* - wrap existing high-value APIs as tools available to LmRaC
 - [GeneCards](#)
 - [GenomicKB](#)
 - [Ensembl](#)
 - [NCBI GEO](#)
 - [ClinicalTrials.gov](#)
- *Generalized RAG* - the ability to interface easily with *any* external data source

CHAPTER 7 SCIENTIFIC DISCOVERY

7.1 Abstract

Scientific discovery is an iterative process that requires transparent reasoning, empirical validation, and structured problem-solving. This work presents a novel human-in-the-loop AI system that leverages case-based reasoning to facilitate structured scientific inquiry. The system is designed to be note-centric, using the Obsidian note-taking application as the primary interface where all components, including user inputs, system cases, and tool specifications, are represented as plain-text notes. This approach ensures that every step of the research process is visible, editable, and revisable by both the user and the AI. The system dynamically retrieves relevant cases from past experience, refines hypotheses, and structures research workflows in a transparent and iterative manner. The methodology is demonstrated through a case study investigating the role of TLR4 in sepsis, illustrating how the system supports problem framing, literature review, hypothesis formulation, and empirical validation. The results highlight the potential of AI-assisted scientific workflows to enhance research efficiency while preserving human oversight and interpretability.

7.2 Introduction

Large language models (LLMs) have the potential to transform scientific research. They offer broad domain knowledge and the ability to synthesize complex information. However, their application in scientific inquiry is hindered by issues such as hallucination, lack of transparency, and difficulty in tracing the reasoning process behind generated insights [138]. To ensure that AI-driven research remains reliable, verifiable, and ethical, human-in-the-loop methodologies are essential.

Here we present a system that integrates case-based reasoning (CBR) [193, 206] with a note-centric workflow to facilitate AI-assisted scientific inquiry. The system is designed around the Obsidian note-taking application (<https://obsidian.md/>) such that all elements of the workflow are represented as first-class plain-text notes in Obsidian. This structure provides a transparent, revisable, and interactive environment where users can inspect, modify, and refine the reasoning process at every stage.

The core workflow of the system follows a structured inquiry process. When a user poses a scientific question or problem, the system assesses whether it aligns with existing case knowledge and retrieves or adapts cases from prior solutions. Importantly, every step of a solution is documented within the note interface, including both user and LLM input, ensuring full traceability. Each step makes use of tools which can be called on explicitly, or searched for based on context.

We demonstrate the capabilities of this approach through a case study exploring the role of TLR4⁷ in sepsis. This example illustrates how the system facilitates problem framing, literature review, hypothesis generation, and data integration. The case study highlights the advantages of this structured, AI-augmented workflow.

The integration of CBR with an interactive note-taking application enhances transparency and trust in problem-solving, fostering a seamless collaboration between human expertise and machine intelligence.

⁷Toll-like receptor 4 (TLR4) plays a central role in detecting bacterial infections. However, in some cases, it can trigger an excessive immune response, leading to sepsis.

7.3 Methods & Design

The system uses a human-in-the-loop approach that is note-centric. That is, all components of the system are stored as notes in the Obsidian note-taking application. All notes are plain-text documents. This includes not only user notes but all system CBR cases as well as tool specifications. This approach means that all elements of the system are transparently available to both the user and LLM as part of the workflow.

7.3.1 System Workflow

Figure 37 gives an overview of the system workflow. The user interacts with Obsidian, the note-taking application. While taking notes, the user may prompt the system to answer a question or solve a problem. The system evaluates the request and searches for any applicable CBR case. A new instance of the most similar case is then created and linked to from the current user note. If no case is found, a default case is created to initiate stepwise problem solving.

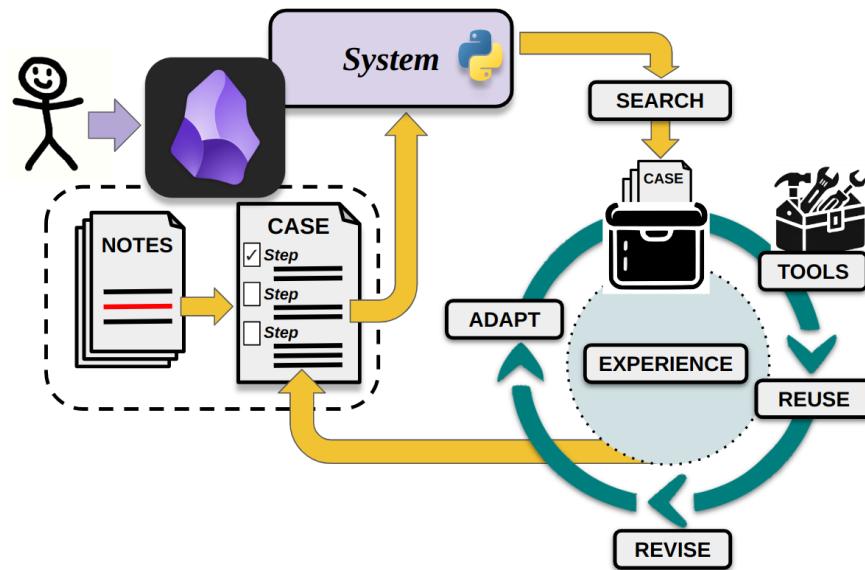


Figure 37: System workflow.

7.3.2 Case-based Reasoning

Case notes are structured documents that encapsulate knowledge for solving problems. Each case includes: a description of the problem, a series of steps for solving the problem and references to optional resources. Steps include an *Action* and may specify pre-conditions (*Requires*). The action is typically composed of a combination of free text instructions and references to system tools. When a tool is executed its response may be included inline in the note, or stored in a context variable. Variables may be passed to later steps. Abstractly, cases represent system experience based on previous problem solving instances. Cases may be reused, revised or adapted as new problems are encountered.

7.3.3 The Collaboration Process

After a case is instantiated and linked to the user's note the system begins execution of the steps. Because the case is a plain-text note, the user sees execution as it progresses. The user may pause execution to review, revise, and/or repeat steps. This keeps the user in-the-loop and makes the reasoning process interactive, transparent and traceable.

7.3.4 Language Enabled Tools

Tool usage and interface is specified in tool notes. As notes, this makes tools searchable both by the user and the system. This means that if a case step specifies some action, the system can search tool specifications for an appropriate tool to perform that action. Tools can also perform language functions (e.g., summarize) as well as retrieve or manipulate data (e.g., from user experiments) through a REST API. This also allows interface to any third-party database.

7.3.5 Implementation

The system is written in Python and interacts with the Obsidian note-taking application through notes written in plain-text markdown. For LLM-based tools Python interfaces to models (GPT4o and o1) through OpenAI’s API [21]. A Pinecone (<https://www.pinecone.io/>) serverless vector database maintains embedding for both search and domain specific document stores.

7.4 Case Study

Given that most available benchmarks assume significant autonomy/agency in performing knowledge discovery tasks [207–209] or focus on a single correct/best answer [210, 211], we instead provide an end-to-end case study to demonstrate how a note-based system facilitates scientific inquiry through *collaboration with a user*. Specifically, the researcher initiates an exploration of how the TLR4 gene is related to sepsis. The approach supports an iterative framework that integrates user input, literature review, external database searches, hypothesis formulation and experimental results. Each step builds upon the previous, ensuring a well-documented and transparent reasoning path that is flexible, adaptable and supports a productive collaboration between human and machine.

7.4.1 Research Question

We begin by adding the following question to a new Obsidian note:

How is TLR4 related to sepsis?

The system must first contextualize the question/problem within the broader framework of scientific inquiry. This helps to set expectations for the nature of the insights and, more importantly, identify appropriate case-based reasoning (CBR) cases relevant to

the question. A summary (Figure 38) is prepared by the *Note Change* case which assesses the original question.⁸

Step: Question Type

Step Completed

- **Type of Question:** Mechanistic and Causal
- **Mechanistic:** The study aims to uncover the biological mechanisms by which TLR4 influences sepsis pathogenesis, such as its role in signaling pathways, immune responses, and downstream effects.
- **Causal:** Investigating how TLR4 activation or inhibition impacts sepsis outcomes, including inflammation severity, organ dysfunction, and survival rates.
- **Related Subtypes:**
 - **Descriptive:** Gathering information about TLR4 expression patterns in sepsis.
 - **Experimental:** Designing experiments to test hypotheses about TLR4's role (e.g., knockdown studies, inhibitor trials).
 - **Predictive:** Using computational models to predict outcomes of targeting TLR4 in sepsis.
- **Relevance to Biological Sciences:**
 - The problem belongs to systems biology and immunology, where understanding signaling networks and their perturbations in disease states is key.
 - It also spans translational research, focusing on identifying therapeutic targets and biomarkers.

Figure 38: Defining the question type and identifying CBR cases (e.g., Mechanistic and Causal questions)

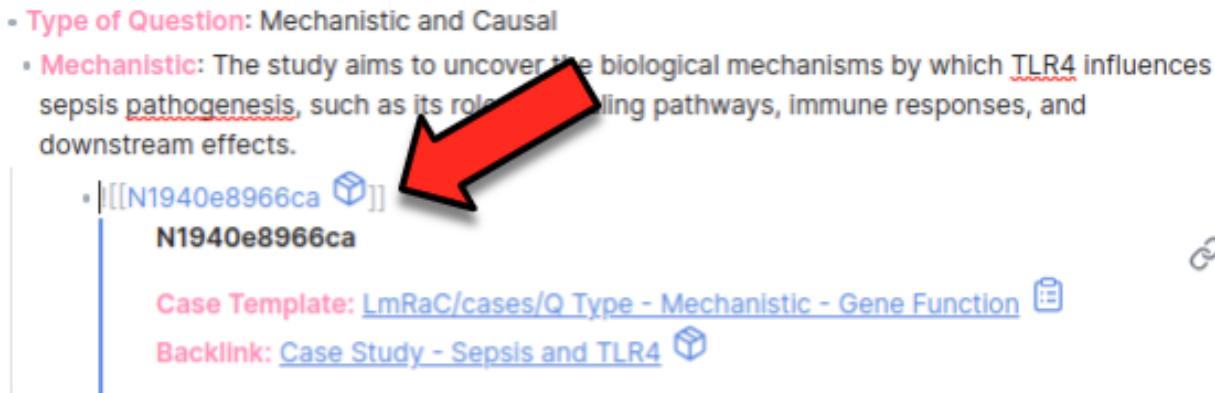
The user reviews the assessment, makes any needed changes and approves the summary (by checking "Step Completed"). This step will act as a guide for subsequent steps. That is, it informs and constrains subsequent step formulation as a part of any reasoning by the LLM. Note, especially, that this text is added to the original user note.

⁸Note that most figure screenshots include red spellcheck underlines from Obsidian.

7.4.2 Initiating Case-based Reasoning

If available, an applicable "Question Type" case is chosen as a top-level starting point (if not, a "default" case formulates one). Here we have two main case types: mechanistic and causal, as well as three subtypes: descriptive, experimental and predictive. The original user note with question (and summary) will act as a top-level note with links to each subsequent reasoning step (Figure 39). The link is not to the Question Type case, but, rather, a copy which will be "populated" as each step in the case is completed and can be edited at any time by the user.

- **Type of Question:** Mechanistic and Causal
- **Mechanistic:** The study aims to uncover the biological mechanisms by which TLR4 influences sepsis pathogenesis, such as its role in signaling pathways, immune responses, and downstream effects.



```

[[N1940e8966ca 📄]] N1940e8966ca
Case Template: LmRaC/cases/Q Type - Mechanistic - Gene Function 📄
BackLink: Case Study - Sepsis and TLR4 📄

```

Figure 39: Instantiate initial CBR case, Mechanistic Gene Function, based on previous experience. Link this (red arrow) to the top-level user note.

Note that steps in the following sections are specific to the above CBR case and, in fact, are only for the "Mechanistic" portion of the question as formulated in Figure 38. Though case specific, each step highlights features of the system available to any case.

7.4.3 Defining Scope

The first step of the 'Mechanistic - Gene Function' case is to define the scope of the problem. This requires user input. The step definition specifies this dependency in the *Requires* section with the instruction: "User input" (Figure 40). From what is known so far

about the problem (which required user approval, see Figure 38) the LLM constructs a list of questions for gathering scoping information.

Step: Define Scope

Step Completed

Requires

User input.

Action

Define scope of problem. Are there elements of the mechanism we are more interested in than others. Are there elements that we are not interested in for this question.

Figure 40: Case reasoning step requiring user input as part of the scope definition action.

Given the task of exploring a gene, this list (Figure 41) asks a series of questions designed to set bounds on what is to be investigated about that gene (e.g., species, interactions, relationship to disease).

User Input

Here's a list of questions to gather additional input before defining the scope of the problem:

1. Focus and Depth

- Are you focusing on human studies, animal models, or both?
 - human only
- Should the analysis be limited to specific tissues or organs (e.g., lungs, liver, blood)?
 - no
- Do you want to prioritize any particular stage of sepsis (e.g., early, late, or recovery phase)?
 - early (presentation to emergency department)

2. Data and Literature

- ignore this as part of the scoping step
- ~~Do you already have access to any datasets, or do you want guidance on identifying relevant datasets (e.g., RNA-seq, proteomics, etc.)?~~
- ~~Should the scope include a review of the latest clinical or experimental findings on TLR4?~~

3. Mechanisms and Pathways

- Are there specific pathways or signaling cascades (e.g., NF- κ B, MAPK) that you want to prioritize in the analysis?
 - no
- Should the role of TLR4 be explored in both pro-inflammatory and anti-inflammatory responses?
 - focus on pro-inflammatory in sepsis

4. Integration with Other Factors

- Do you want to explore TLR4 interactions with specific co-factors or receptors (e.g., MD-2, CD14)?
 - not sure
- Should the analysis include the microbiota or other external factors influencing TLR4 activation?
 - yes, definitely

5. Sepsis Context

- Do you want to differentiate between the role of TLR4 in bacterial, fungal, or viral sepsis?
 - focus on bacterial
- Should the analysis explore the genetic polymorphisms of TLR4 and their association with sepsis outcomes?
 - no

Figure 41: Requested user information for scope definition. User interleaves answers and then checks boxes. User may also use strike-through to indicate information that should be ignored (i.e., is out of scope).

Key to the system's collaborative design is that this is *not* a passive solicitation of information from the computer. Rather, the user may edit the list in any way, including using strikethrough to signal that items should be ignored. User answers are interleaved with questions and checkboxes are used to indicate the user has completed the question.

Scope Definition

The study will focus on understanding the biological mechanisms by which TLR4 influences early sepsis pathogenesis in humans, particularly during the presentation to the emergency department. The analysis will explore the pro-inflammatory role of TLR4 in sepsis, focusing specifically on bacterial sepsis.

The study will not be limited to specific tissues or organs, aiming for a systemic perspective of TLR4 activation. It will include external factors such as the microbiota that influence TLR4 activation and their potential role in sepsis development. While the role of TLR4 interactions with specific co-factors or receptors (e.g., MD-2, CD14) remains uncertain, this may be considered if it emerges as critical during the investigation.

The scope excludes the exploration of genetic polymorphisms of TLR4 and downstream signaling pathways (e.g., NF-κB or MAPK), focusing instead on its broader pro-inflammatory activity and systemic impact during bacterial sepsis.

Figure 42: Final system composed question scope definition.

Given the user feedback, the Action portion of the step proceeds and the system proposes a working definition of scope (Figure 42). Again, the user may edit and revise as appropriate since all text is part of an Obsidian note. At this point the step is checked as completed and the next step begins.

7.4.4 Refining the Problem

We are now in a position to refine the original problem statement based on the agreed upon scope. In this step the Action implicitly uses the LLM to generate a Refined Problem Statement and propose Key Questions. The Step definition and results are both given in Figure 43.

Step: Refine Problem

Step Completed

Action

Using the question scope definition: refine the original problem statement to focus on a few key questions.

Refined Problem Statement

The study aims to uncover the biological mechanisms by which TLR4 influences early sepsis pathogenesis in humans, focusing on its pro-inflammatory role during bacterial sepsis. The analysis will explore systemic effects and external factors, such as the microbiota, that influence TLR4 activation at the time of emergency department presentation.

Key Questions:

1. How does TLR4 activation contribute to the systemic pro-inflammatory response observed in early bacterial sepsis?
2. What role do external factors, including the microbiota, play in modulating TLR4 activation during bacterial sepsis?
3. Are there specific mechanisms of TLR4-mediated inflammation that can be targeted to improve outcomes in early sepsis?
4. How does the timing and magnitude of TLR4 activation influence the progression of early bacterial sepsis, and what are the critical thresholds for transitioning from a controlled to a dysregulated immune response?

Figure 43: Refined problem statement and key questions. An additional question has been added by the user (blue highlight).

Notice that the user has exercised the option of adding an additional question (blue highlight). More generally, the system also fully supports not only editing responses, but also the Action definition itself. This serves two purposes, improving responses for a particular problem step, but also providing a mechanism for system learning. Since all case instances represent experience, any changes in how a problem is approached becomes an opportunity to refine and adapt CBR cases for future problems.

7.4.5 Quick Review

The adoption of LLMs for scientific research has been hindered by, among other things, their propensity to fabricate both information and citations supporting those fabrications [212]. Nevertheless, their breadth of training can make them invaluable partners *if verification is included*.

In this step the LLM is used to provide a quick (though potentially unreliable) review of the problem. The Action uses an explicit system tool call to effect a search given the

previously generated Problem Statement (Figure 44).

Step: Quick Review

Step Completed

Action

Ask GPT for a review of the Refined Problem statement.

```
res0 = tool:GPTsearch({query})
{res0["answer"]}
```

Figure 44: Explicit tool call to ask OpenAI o1 model for "quick" unauthoritative answer.

By making the call explicit the return response can be captured in a named variable, *res0*. In doing so, the response will not be included as part of the note, however, by using a substitution statement after the tool, `{res0["answer"]}`, the response (aka answer) is both included in the note text (Figure 45) and now stored in a variable for later use (see 7.4.7 below).

1. How does TLR4 activation contribute to the systemic pro-inflammatory response observed in early bacterial sepsis?

TLR4 detects pathogen-associated molecular patterns—especially lipopolysaccharide from Gram-negative bacteria—triggering a cascade that rapidly amplifies the release of pro-inflammatory mediators (e.g., TNF- α , IL-1 β , and IL-6). This surge in cytokines and chemokines mobilizes immune cells and activates endothelial pathways, leading to widespread vascular permeability, tissue injury, and the hallmark systemic inflammatory response of sepsis.

2. What role do external factors, including the microbiota, play in modulating TLR4 activation during bacterial sepsis?

The gut microbiota can prime or dampen TLR4-mediated responses by influencing baseline immune surveillance and shaping local inflammation in barrier sites. Changes in microbial composition or permeability (as in dysbiosis) can either heighten TLR4 signaling by exposing the host to more bacterial products or restrain it by promoting beneficial commensal interactions and regulatory immune pathways.

3. Are there specific mechanisms of TLR4-mediated inflammation that can be targeted to improve outcomes in early sepsis?

Targeting downstream signaling molecules, such as MyD88 and TRIF, or blocking TLR4 directly can modulate the excessive inflammation without fully compromising pathogen clearance. Novel therapeutic strategies also focus on interrupting the downstream cytokine storm or stabilizing the endothelium, aiming to reduce the deleterious effects of hyperinflammation while preserving essential immune defenses.

4. How does the timing and magnitude of TLR4 activation influence the progression of early bacterial sepsis, and what are the critical thresholds for transitioning from a controlled to a dysregulated immune response?

Early and balanced TLR4 engagement is crucial for effective pathogen recognition and clearance. However, if TLR4 is activated too robustly or persists beyond an initial window, the immune response can become dysregulated, shifting from protective to harmful. This tipping point often involves excessive cytokine release, endothelial dysfunction, and organ injury, signifying a transition from a contained inflammatory response to full-blown septic pathology.

Figure 45: "Quick" answers to the refined problem's key questions.

The LLM (OpenAI o1 in this case) answers each Key Question in a plausible, though unverified, manner. This step is meant to offer the user a quick overview as orientation to the problem space as well as prompt revision of earlier steps if the user believes this is appropriate (e.g., answers do not support a hypothesis the user has in mind).

7.4.6 Literature Review

With this cursory look at the answer in mind we may now undertake a more rigorous literature review. In this step the Action explicitly requests a search of PubMed articles to answer the Key Questions (Figure 46). Key to this tool is its design to explicitly *not* answer questions using the LLM. Instead, it collects articles based on a keyword search of PubMed (e.g., genes, diseases, pathways), indexes them and uses retrieval-augmented generation (RAG) to answer the questions [48].

Step: Literature Review

Step Completed

Action

Perform a PubMed search for each Key Question of the Refined Problem statement.

```
res1 = tool:PubMedSearch({query})
{res1["answer"]}
tool:summarizeRefs({res1["references"]})
```

Figure 46: Initiate a literature search of PubMed on the list of Key Questions. Search results are stored in *res1* and references for search results are passed to another tool to be summarized.

Answers include paragraph-level citations to all articles used to answer the question as well as a complete linked bibliography (Figure 47).

QUESTION

How does TLR4 activation contribute to the systemic pro-inflammatory response observed in early bacterial sepsis?

ANSWER

TLR4 activation plays a critical role in the systemic pro-inflammatory response observed in early bacterial sepsis by recognizing and binding to lipopolysaccharides (LPS), components of the outer membrane of Gram-negative bacteria. This binding triggers signaling pathways, primarily through the MyD88-dependent and TRIF-dependent pathways, leading to the activation of nuclear factor kappa B (NF- κ B) and mitogen-activated protein kinases (MAPKs) ([Shen, X. et al., 2024 ↗](#)) ([Qiu, F. et al., 2023 ↗](#)). These pathways result in the production and release of pro-inflammatory cytokines such as TNF- α , IL-1 β , and IL-6, key mediators of inflammation ([Punch, E. et al., 2022 ↗](#)) ([Qiu, F. et al., 2023 ↗](#)) ([Jeon, D. et al., 2024 ↗](#)). The excessive release of these cytokines contributes to the systemic inflammatory response syndrome (SIRS) and can lead to multiorgan failure, a hallmark of sepsis ([Fernández-Martín, JC. et al., 2022 ↗](#)) ([Park, YJ. et al., 2023 ↗](#)). Additionally, TLR4 activation can lead to endothelial dysfunction and coagulopathy, further exacerbating the inflammatory response and contributing to the pathogenesis of sepsis ([Kuzmich, NN. et al., 2017 ↗](#)) ([Perrin-Cocon, L. et al., 2017 ↗](#)).

REFERENCES

Fernández-Martín, JC; Espinosa-Oliva, AM; García-Domínguez, I; Rosado-Sánchez, I; Pacheco, YM; Moyano, R; Monterde, JG; Venero, JL; de Pablos, RM. Gal3 Plays a Deleterious Role in a Mouse Model of Endotoxemia. International journal of molecular sciences. 2022. [PMC8835800 ↗](#)

Jeon, D; Hill, E; McNeil, DG. Toll-like receptor agonists as cancer vaccine adjuvants. Human vaccines & immunotherapeutics. 2024. [PMC10760790 ↗](#)

Kuzmich, NN; Sivak, KV; Chubarev, VN; Porozov, YB; Savateeva-Lyubimova, TN; Peri, F. TLR4 Signaling Pathway Modulators as Potential Therapeutics in Inflammation and Sepsis. Vaccines. 2017. [PMC5748601 ↗](#)

Park, YJ; Seo, KH; Joo, JD; Jung, HS; Kim, YS; Lee, JY; Park, H. The effects of etomidate on expression of high mobility group box 1 via the nuclear factor kappa B pathway in rat model of sepsis. The Libyan journal of medicine. 2023. [PMC9980160 ↗](#)

Perrin-Cocon, L; Aublin-Gex, A; Sestito, SE; Shirey, KA; Patel, MC; André, P; Blanco, JC; Vogel, SN; Peri, F; Lotteau, V. TLR4 antagonist FP7 inhibits LPS-induced cytokine production and glycolytic reprogramming in dendritic cells, and protects mice from lethal influenza infection. Scientific reports. 2017. [PMC5247753 ↗](#)

Punch, E; Klein, J; Diaba-Nuhoho, P; Morawietz, H; Garelnabi, M. Effects of PCSK9 Targeting: Alleviating Oxidation, Inflammation, and Atherosclerosis. Journal of the American Heart Association. 2022. [PMC9238481 ↗](#)

Qiu, F; Zeng, C; Liu, Y; Pan, H; Ke, C. J147 ameliorates sepsis-induced depressive-like behaviors in mice by attenuating neuroinflammation through regulating the TLR4/NF- κ B signaling pathway. Journal of molecular histology. 2023. [PMC10635911 ↗](#)

Shen, X; He, L; Cai, W. Role of Lipopolysaccharides in the Inflammation and Pyroptosis of Alveolar Epithelial Cells in Acute Lung Injury and Acute Respiratory Distress Syndrome. Journal of inflammation research. 2024. [PMC11370780 ↗](#)

Figure 47: Literature Review answer to first Key Question. Answer is based *only* on PubMed articles with paragraph-level links to citations and bibliography [4–11].

Though not shown here, the user has the option of revising any question, asking additional questions or otherwise annotating these results.

7.4.7 Knowledge Gaps

Having completed a first review of the Key Questions, we can now attempt to identify knowledge gaps that may warrant further investigation. This step (Figure 48) uses the same GPT tool as before but now incorporates information from previous steps (7.4.5 and 7.4.6) as part of the prompt using variable substitution (red arrows). It also explicitly specifies the structure of the response using a function prototype. Although this may border on "programming" for many users, it is shown here to demonstrate the level of control a user has over *how* the LLM answers questions.

Step: Knowledge Gaps

Step Completed

Action

```
{"params":"
FUNCTION assessKnowledgeGaps 'Assess knowledge gaps in the literature'
PARAMETER knowledgeGapList:ARRAY    "List of gaps"
|   ITEM gap:STRING    "Description of open question (gap)"
|
}

res2 = tool:GPT(
"You are an intelligent agent that assesses complex scientific questions. Given the following literature review, what
knowledge gaps do you see that might warrant further investigation. Take into account the potential difficulty as well as
the potential reward of investigation."
"""Summary Over
{res2["answer"]}
Literature Review:
{res1["answer"]}
"""
{params}
)
```



Figure 48: GPT API is used to assess knowledge gaps. An explicit prompt is provided along with results from the previous Quick Review and Literature Review. The response is structured using a function prototype parameter.

The LLM, using both reviews, then provides a list of nine potential gaps in knowledge (Figure 49 shows the first two).

1. Timing and Biomarker Precision for TLR4 Activity

- **Gap:** Although timing and magnitude of TLR4 activation are emphasized as critical, precise biomarkers or methods to measure real-time TLR4 activity in early sepsis remain vague. It is unclear which surrogate markers (e.g., cytokines, soluble TLR4, or downstream signaling molecules) most reliably indicate when TLR4 signaling has transitioned from beneficial to harmful.
- **Difficulty:** Moderately high, as it requires careful study design (e.g., serial measurements in emergency department patients) and potentially the development of novel assays.
- **Reward:** High, because reliable biomarkers could guide early interventions and identify the optimal window for TLR4-targeted therapies.

2. Microbiota Composition and Site-Specific Influences

- **Gap:** While gut dysbiosis is discussed as a driver of elevated LPS and TLR4 activation, the exact strains or metabolites responsible, along with the regional differences in gut microbiota (e.g., small intestine vs. colon), are poorly defined. Similarly, less is known about how the lung microbiome or oral microbiome might shape TLR4 activation in early sepsis.
- **Difficulty:** High, given that multi-omics approaches (16S rRNA sequencing, metabolomics) and integrative data analyses are required.
- **Reward:** Potentially transformative. Clarifying these microbiota-TLR4 linkages could open avenues for precision probiotic/prebiotic interventions or dietary modifications to minimize hyperactivation of TLR4.

3. TLR4 Crosstalk with Other Receptors and Signaling Pathways

- **Gap:** TLR4 seldom acts in isolation—crosstalk with TLR2, TLR9, NOD-like receptors (NLRs), and RIG-I-like

Figure 49: First two identified knowledge gaps (of nine) summarized and annotated by difficulty and reward.

Again, the strength of the LLM to identify and summarize is leveraged to provide a concise annotated summary of each potential gap. It remains up to the user to review, refine and approve the results. However, the system, by design, provides a documented, transparent path of reasoning steps to assist in this task.

7.4.8 Database Review

In addition to GPT and PubMed reviews, the user can also incorporate knowledge from other sources. This step (Figure 50) demonstrates the use of system tools to search, summarize and structure information from external sources, in this case: Wikipedia and GeneCards.

Step: Database Review

Step Completed

Action

```
{"dbList":["Wikipedia","GeneCards"]}
res3 = tool:dbSearch("TLR4",{dbList})
{res3}
```

Wikipedia: TLR4

General Characteristics

- **Name:** Toll-like receptor 4 ([TLR4](#)), also known as [CD284](#).
- **Function:** Activates innate immunity and plays a key role in bacterial infection defense.
- **Structure:** Transmembrane protein (~95 kDa) encoded by the TLR4 gene.
- **Expression:**
 - High in myeloid immune cells (e.g., monocytes, macrophages, dendritic cells).
 - Low in some non-immune cells (e.g., epithelium, endothelium, placental cells, Langerhans islet beta cells).

Activation and Ligands

- Recognizes pathogen-associated molecular patterns (PAMPs) such as [lipopolysaccharides \(LPS\)](#) from Gram-negative bacteria.
- Recognizes damage-associated molecular patterns (DAMPs) like HMGB1, S100 proteins, and histones, which are released during tissue damage or cell death.
- Facilitated by:
 - LPS-binding protein (LBP)
 - CD14
 - MD-2 protein (essential for receptor signaling).

Figure 50: Step definition to use two external data sources (Wikipedia and GeneCards) to summarize more information about the TLR4 gene.

The *dbSearch* tool takes a list of sources, queries them and then returns them to the user note. As before, these are added to the note, but also saved to a variable, *res3*, for later use by other functions.

7.4.9 Hypothesis Formulation

Given our Problem Statement; literature and database reviews; and, assessment of knowledge gaps, we can now attempt to formulate reasonable research hypotheses (Figure 51).

Step: Hypothesis Formulation

Step Completed

Action

For the refined problem, formulate three hypotheses for further investigation using the above literature review, database review and knowledge gaps.

Figure 51: Using the results of multiple previous steps: formulate research hypotheses.

Reiterating: all work to this point has been recorded in a single user readable/editable note in Obsidian. It is available to the user, but also to the system. The Action in this step takes advantage of the accumulation of knowledge to formulate the hypotheses (Figure 52).

Hypothesis 2: Gut Microbiota-Driven Modulation of TLR4

Hypothesis

Dysbiosis of the gut microbiota—characterized by specific strain imbalances and elevated microbial-derived lipopolysaccharide (LPS)—amplifies TLR4 signaling in the early phase of bacterial sepsis, thereby exacerbating systemic inflammation and worsening patient outcomes.

Rationale

- Directly tackles the gap on microbiota composition and site-specific influences (Gap #2).
- Integrates evidence that gut-derived LPS is a major activator of TLR4, with dysbiosis increasing intestinal permeability and systemic inflammatory responses.
- Extends current knowledge on how external factors, such as diet or the microbial environment, drive TLR4-mediated sepsis.

Potential Study Approach

1. Microbiome profiling (16S rRNA sequencing) of stool samples from sepsis patients at ED admission and 24 hours later.
2. Metabolomic analysis of short-chain fatty acids and other microbial byproducts known to influence TLR4 signaling.
3. Correlation of microbiota composition and metabolite levels with TLR4-associated biomarkers (e.g., NF-κB activation, IL-6 levels).
4. Interventional arm testing whether restoring gut microbiota balance (via targeted probiotics or fecal microbiota transplantation) reduces TLR4 hyperactivation and improves clinical outcomes.

Figure 52: Proposed Hypothesis 2, including: rationale and potential stepwise approach.

Three hypotheses are stated. The rationale for each is summarized and potential approaches for their study are given. This step in particular is a starting point. A user is expected to iterate and refine a hypothesis. This may mean qualifying or constraining

a given hypothesis and re-running, or it may involve returning to earlier steps to gather more information (e.g., literature review). The use of a note taking system is meant to encourage and support the dynamic collaboration that is key to a scientific workflow.

7.4.10 Experiments and Data Collection

Another key feature of the system is the ability to seamlessly incorporate external data into reasoning tasks. In this case study the user has indicated an interest in Hypothesis 2 which integrates gut microbiota with changes in TLR4 signaling during sepsis (Figure 52). The Action has been edited by the user to focus data source search on this hypothesis (Figure 53).

Step: Experiments / Data Collection

Step Completed

Action

What experiments or existing data sources would be useful for the proposed hypothesis number 2? Specifically, data that would potentially help provide some preliminary data for a larger project.

Figure 53: *Implicit* search of external resources for datasets suitable for preliminary results.

This search utilizes another section of the CBR case: *Suggested Resources* (Figure 54). This gives the system an *implicit* starting point for finding relevant data. Note that initially the databases are not themselves searched, but, rather, the LLM (OpenAI o1) utilizes its own training to locate possible sources. Like the GPT Literature Review (see 7.4.5) this is not meant to be a final authoritative search. Rather, it quickly locates possible data as well as giving guidance to the user (not shown) on how to search the database resources (e.g., GEO and SRA).

Suggested Resources

% tools and other resources given priority for this particular case
% especially those that have proven useful in past experience

Tools

- [iPathwayGuide](#)
- [DESeq2](#)

Databases

- [GenomicKB](#)
- [NCBI Gene Expression Omnibus \(GEO\)](#)
- [NCBI NCBI Sequence Read Archive \(SRA\)](#)

Figure 54: The CBR case includes Suggested Resources. This includes one for gene expression (GEO) and one for microbiome profiling (SRA).

Excerpted search results for both gene expresion (from GEO) and microbiota profiling (SRA) are given in Figure 55. These results include assession identifiers (red arrows) as well as descriptions and relevance for Hypothesis 2 use.

2. Example GEO Datasets Related to Sepsis & Microbiota

Below are a few datasets that, while not perfect "one-stop shops" for Hypothesis 2, may still provide useful preliminary data. They either contain (a) gene expression data for sepsis and TLR4-related pathways or (b) microbiome-related data in the context of inflammation.

1. GSE65682

- **Study Type:** Whole blood transcriptome profiling in septic shock patients.
- **Relevance:** Provides a large sample size for examining *host inflammatory gene expression*, including TLR4 pathways.
- **Limitations:** Does *not* include stool microbiome. You could, however, explore whether TLR4 expression correlates with outcome or disease severity.

2. Selected Example Datasets from SRA/ENA

Below are a few examples that come up in searches when looking for **16S rRNA or metagenomic data in sepsis contexts**. Depending on your focus, some may be closer matches to **Hypothesis 2** (i.e., gut microbiome involvement in TLR4 signaling) than others.

A. Human Sepsis Gut Microbiome Datasets

1. BioProject: PRJNA530371

- **Description (as listed):** 16S rRNA sequencing of fecal samples from patients with sepsis and healthy controls.
- **Relevance:** Provides gut microbiota profiles that may be linked with inflammatory status or clinical outcomes.
- **How to Access:**
 - NCBI SRA: [PRJNA530371](#)
 - Check if the associated publication or metadata mention TLR4 expression or host transcriptomic data in parallel.

Figure 55: Excerpts from the search using OpenAI o1 model. Public datasets are identified (red arrows). Searches may also be done using tools to directly access resource APIs.

Again, this is meant as a way of using the LLM to quickly assess the availability of relevant public datasets. The user may then utilize other system tools (not shown) to search and download the actual data from GEO and SRA.

7.4.11 Differential Expression Analysis

Given that most data analysis, especially in the biological sciences, involves a multi-step pipeline, the advantages of initiating and monitoring a pipeline from a notes interface

are limited. However, the system *does* have access, via its built-in REST API interface, for accessing the results of any analysis. What this means practically, is that these results can be incorporated into the workflow like any other text source.

Step: Differential Gene Expression Analysis

Step Completed

Action

Analyze experimental results to find differentially expressed genes.

Results are from DESeq2.

Figure 56: *Explicit* tool identification for computing differential gene expression on retrieved datasets.

Figure 56 implicitly calls a tool to interpret the results of a standard DESeq2 differential gene expression analysis. In this case the tool expects a list of genes in CSV format that includes: gene symbol (e.g., TLR4), log fold-change and significance of the change (typically, adjusted p-value). The results of the step are to summarize those genes that have been found to be significantly differentially expressed.

The user may then qualify these results relative to the workflow by posing additional questions (e.g., "Are other genes associated with an inflammatory response also up-regulated?").

7.4.12 Experimental Insights and Reflection

LLMs are particularly adept at summarization tasks. This CBR case takes advantage of this feature and asks the LLM in this final step to reflect on what has been discovered thus far and to suggest next steps (Figure 57).

Step: Insights

Step Completed

Action

What insights can we extract from the pathway analysis results.

Outline next steps for investigation.

Figure 57: Preliminary results analysis and assessment of next steps sensitive to this analysis (if results are available).

Recall that each step has the accumulated context of all previous steps, so, although the Action may seem vague by asking for "insights," it is actually operating on the accumulated text of everything that has come before (nearly 20 pages in this case study).

In contrast to many autonomous reasoning systems (e.g., OpenAI's o1 model), this system is designed to support a scientific workflow that is *expected* to be open-ended and subject to continuous revision as hypotheses are generated and experiments are performed and interpreted. There is rarely, if ever, one right answer.

Figure 58 supports this workflow not only with a summary and critique of foregoing steps, but by giving guidance for refining and extending the work.

7.5 Discussion

The case study presented here illustrates how a human-in-the-loop AI system can enhance the process of scientific discovery. By structuring inquiry through case-based reasoning, the system provides a transparent, traceable, and iterative approach that naturally aligns with standard scientific workflows. A key strength of this approach is its ability to leverage LLMs as an integral tool for productive human collaboration.

A critical challenge in leveraging AI for scientific discovery is ensuring that the generated insights remain grounded in empirical evidence. LLMs are known to generate

1. Refine Hypotheses and Validate Biomarkers

1. Synthesize Findings

- Integrate the preliminary insights (e.g., correlations between specific microbial taxa and host TLR4-related genes) with existing literature.
- Identify which bacterial taxa, metabolites, or gene signatures emerged as most strongly associated with TLR4 signaling in sepsis.

2. Define Candidate Biomarkers

- From your data-driven correlations, pick the top microbial markers (e.g., certain Gram-negative strains or LPS-related metagenomic signatures) and host immune markers (e.g., TLR4, NF-κB, IL-6) that show promise as early indicators of dysregulated TLR4 activity.
- Confirm feasibility of assaying these markers (e.g., qPCR for bacterial strains, ELISA for cytokines).

3. Develop Targeted Assays

- If new assays or optimized pipelines are needed for quantifying TLR4 or gut microbes in real time (e.g., multiplex PCR, digital droplet PCR), plan or pilot these methods using clinical or model system samples.

2. Conduct Pilot Prospective or Controlled Studies

1. Small-Scale Human Cohort

- Design a pilot prospective study in an emergency department (ED) setting:
 - Recruit patients with suspected sepsis at presentation.
 - Collect paired blood and fecal samples to measure both TLR4-driven inflammation (e.g., soluble TLR4, cytokines) and gut microbiome composition.
 - Time Points: ED admission, 6–12 hours later, and ~48 hours later to capture early sepsis progression.

2. Murine or Preclinical Models

- Use a controlled animal model (e.g., cecal ligation and puncture, CLP) to directly test causality between gut dysbiosis, TLR4 activation, and sepsis severity.
- Manipulate gut microbiota (via antibiotics, probiotics, or fecal transplants) and assess changes in TLR4 expression, cytokine profiles, and clinical outcome (e.g., survival, organ injury).

3. Data Analysis & Validation

- Compare pilot findings with public datasets to see if observed relationships (e.g., high abundance of certain LPS-producing bacteria correlating with TLR4 overactivation) are consistent across cohorts.

Figure 58: First two (of six) suggested next steps for the investigation including, hypothesis refinement, validation and prospective controlled studies.

plausible yet unverified statements, which can mislead researchers if used uncritically. This system mitigates such risks by explicitly incorporating verification steps, including literature searches using PubMed and database reviews via other trusted sources. The interactive nature of the system ensures that the user remains an active partner in refining problem definitions, verifying outputs, and shaping hypotheses. This stands in stark contrast to many recent autonomous-blackbox approaches to LLM reasoning.

The foregoing case study demonstrates the value of structuring problem-solving through an evolving CBR system. Cases represent human-machine experience and as such can be reused, refined and adapted for new problems. Their implementation as first-class notes ensures transparency and encourages human collaboration as part of the reasoning process. In this example the iterative approach to scope definition, literature review, and hypothesis refinement steps serve as checkpoints, reinforcing scientific rigor while allowing for flexibility and refinement in inquiry. Using an LLM to help identify knowledge gaps and synthesize insights from multiple sources highlights the strength of this approach and demonstrates how AI can enhance, rather than replace, the natural reasoning process of scientific experts. Providing a mechanism for retrieving user experimental results further enhances the workflow by facilitating a seamless transition from hypothesis generation to empirical validation.

By embedding this approach within a human note-taking system, LLM-based tools become an integral component of the workflow, fostering a continuous cycle of learning and adaptation driven by user-machine collaboration. Furthermore, storing *all* CBR cases, tools, and generated results as user notes enhances transparency and traceability, ensuring that each step in the reasoning process remains accessible for review and refinement.

7.6 Conclusion

Our approach underscores the potential for human-in-the-loop AI systems to enhance scientific discovery by structuring inquiry, verifying insights, and integrating empirical data. By leveraging case-based reasoning, the approach ensures that LLM-generated outputs remain contextually relevant, empirically grounded, and are subject to a continuous step-by-step review by a collaborating human user.

The results demonstrate that while LLMs provide valuable breadth and summarization capabilities, their true scientific utility emerges when coupled with a human-in-the-loop. The interplay between user expertise and LLM-based tools creates a workflow that is not only transparent and accountable, but also adaptable to the evolving nature of all scientific inquiry. Ultimately, this approach represents a step toward AI-assisted research frameworks that align with the principles of scientific rigor and iterative discovery, paving the way for more effective collaboration between AI systems and domain experts in the pursuit of knowledge.

CHAPTER 8 DISCUSSION

LmRaC demonstrates how incorporating an LLM as a reasoning engine within a larger, transparent, and user-driven architecture can advance the fields of computational biology, scientific discovery, and general AI reliability. By prioritizing user collaboration, this research addresses both the practical demands of data-intensive research and the broader need for ethically sound, trustworthy, transparent and adaptable AI systems.

8.1 Results

8.1.1 Performance

The system described in Chapters 4, 5, 6 and 7 – collectively referred to as LmRaC – demonstrates that large language models can be integrated as sophisticated “language reasoning engines” rather than treated naively as knowledge repositories. Across the three development stages – Command Line, Web Application, and Collaborative Notes – the system reliably answers domain-specific questions by leveraging materials provided by its retrieval-augmented generation modules. This approach effectively minimizes hallucinations and fabricated references.

Accuracy and Traceability

By restricting LLM output to user-curated knowledge (e.g., PubMed full-text, domain documents, or user experimental data), LmRaC provides answers traceable to paragraph-level citations. These results outperform both baseline LLMs (GPT-3.5, Llama 2, PaLM) and a domain-specialized model (BioGPT) in pairwise comparisons, with LmRaC matching or exceeding GPT-4’s accuracy on many scientific queries; this using only domain-knowledge.

Iterative Reasoning

In the Web Application release, iterative prompts allow multi-step reasoning for multi-step biomedical questions, such as those involving differential gene expression experiments. The Collaborative Notes integration extends this capability, enabling an ongoing back-and-forth between user and system on a foundation of case-based reasoning.

User-in-the-loop

A central goal was to ensure the user remained part of the reasoning process, providing oversight and making the user a true agent in the reasoning process. Qualitative feedback suggests that this user-centered design fosters greater trust and a sense of collaboration in problem-solving.

8.1.2 Unexpected Findings

Utility of Usefulness Metric

One of the surprising yet beneficial outcomes is the effectiveness of a two-pass approach (semantic similarity and LLM-based “usefulness” scoring) in filtering relevant document chunks. This significantly boosts answer quality and reduced cost by cutting down on the number of tokens processed.

Over-Answering Simple Questions

In some comparisons, LmRaC’s final answers were occasionally assessed as having excessive detail. Ironically, the iterative process, while beneficial for complex inquiries, tends to over-answer more narrower queries with straightforward answers. This indicates a need for dynamic control of question complexity.

8.1.3 Contextualization

The results align with the broader literature on RAG systems, supporting the notion that domain specificity and user-defined knowledge bases lead to more reliable outputs. Unlike pure "chat" systems that rely on a model's internal parameters, LmRaC focuses on controlled, transparent retrieval. These findings add to the growing body of work showing that augmenting LLMs with curated external knowledge can reduce hallucinations and produce domain-accurate results comparable to even the most advanced GPT models.

8.2 Strengths and Limitations of the Program

8.2.1 Unique Capabilities and Strengths

Multi-tiered RAG Architecture

LmRaC's use of multiple RAGs for: domain knowledge (RAG_{dom}), experimental data (RAG_{exp}), and function interface (RAG_{fun}) allows the system to integrate multiple, specialized data sources, while delivering paragraph-level citations to authoritative sources and seamlessly integrating quantitative user data through a REST API.

Case-Based Reasoning in Collaborative Notes

By treating both cases and tools as first-class notes, LmRaC promotes transparency in how solutions are generated and adapted. This approach bridges the gap between pure LLM chat interfaces and traditional manual scientific workflows.

Explainability and Trustworthiness

Every piece of knowledge can be traced back to the original journal paragraph or user data, minimizing hallucinated content. The system's "usefulness" metric further filters out potentially irrelevant information.

8.2.2 Limitations

Generality vs. Specificity

Although the program handles scientific queries well – particularly in biology and biomedical research – its performance in other domains may require building new or different domain indexes and vocabularies.

Dependency on External APIs

The current design depends on OpenAI's GPT-4 and Pinecone vector databases. Any service disruptions, rate limits, or changes in pricing can affect the system's scalability and uptime.

User-Driven Knowledge Bases

While a user-in-the-loop architecture minimizes hallucinations, it demands some expertise from end-users. As such it is not designed to tutor an inexperienced user in the domain. The system currently does not "discover" new research without explicit user guidance, which can be seen as a trade-off between safety/traceability and automation.

Complex Questions

As questions grow more elaborate, the sequential sub-question approach used by the LmRaC web application may become cumbersome. The risk is either “over-answering” small questions or under-specifying large, deeply interlinked problems. Specifically, questions that require more than a linear accumulation of answers are not suitable for this version.

8.3 Technical Challenges and Solutions

8.3.1 Challenges Encountered

Hallucinations and Citation Fabrication

A primary reason for adopting retrieval-augmented generation and filtering out the LLM's implicit knowledge was to sidestep known issues with hallucination. Yet, ensuring the LLM does not leak "unverified" statements remains a challenge. This is a double-edged sword in that poor quality user information presents the same problem as hallucinations: it can lead to incorrect conclusions.

Performance vs. Usability

Iterative prompting in the LmRaC web application can result in overly verbose or misleading answers. Without some "executive" to manage planning, answers sometimes drift during reasoning. Also, though giving the user a role in answering is a design goal, it also demands some level of participation versus the blackbox approach of many recent advanced models.

Index Confusion

LmRaC command-line, in particular, demonstrated that keeping domain knowledge separated in independent indexes has advantages, but can lead to confusion when no answer is forthcoming due to incorrect index choice. Likewise, the LmRaC web application, while providing an interface to indexes, can result in information overload for the user.

8.3.2 Solutions and Trade-offs

Usefulness Screening

Pre-filtering chunks via the usefulness metric demonstrably improved final answer quality but added an additional LLM call. This trade-off was deemed acceptable because of improved quality.

Sub-questions vs. Single Question

A user-configurable answer complexity setting balances thoroughness against cost and speed though it depends on GPT to create relevant sub-questions that may at times be redundant, especially for simple questions.

Index Separation

Separating domain (RAG_{dom}) and experiment (RAG_{exp}) indexes maintained a boundary between general authoritative knowledge and experiment-specific context. Though it adds complexity, it prevents unintentional mixing of “second-hand” user data with authoritative domain knowledge.

8.4 Validation and Evaluation

8.4.1 Reliability and Accuracy

In controlled pairwise evaluations, LmRaC matched or surpassed large LLMs not augmented with external knowledge on complex biomedical questions. The user-driven approach especially shone when the question invoked recently published data or user-specific experimental results – scenarios where “pure LLM” solutions tend to falter due to outdated training.

8.4.2 Testing Methods and Outcomes

To validate the system's accuracy and reliability, a range of curated scientific questions – some from real-world differential gene expression studies, some from standard scientific Q&A sets – were posed to LmRaC and to standalone LLMs.

8.4.3 Benchmark Testing

While no large systematic test of LmRaC was performed, anecdotal testing *was* used to assess the user-in-the-loop approach and as an aid to design and usability. This included custom curated questions for LmRaC command-line as well as third-party benchmarks like GPQA (Google-Proof Q&A) Benchmark [197] and DiscoveryBench [198] for data-driven discovery while testing the LmRaC Collaborative Notes version. Part of the problem with any comparison of performance with other models is that LmRaC is, by design, human-centric. One might object that the performance of LmRaC on *any* problem is more a function of the *user* than the system given that the user can suggest, revise or otherwise intervene in any reasoning task. *The goal of LmRaC is to improve human performance in collaboration with LLM based technology* **not** design a standalone system to answer PhD-level questions (cf. OpenAI o1 and o3).

8.4.4 Case Studies

Recent LmRaC research [16] has utilized an end-to-end case-study approach for evaluating performance and usability (7.4). Real-world problems (e.g., "How is TLR4 related to sepsis?") are used to initiate a full collaboration between user and system. This includes: defining scope, refining the problem, literature review, knowledge gap identification, hypothesis formulation and experimental review and design.

8.5 Broader Implications

The design principles behind LmRaC – particularly the emphasis on user-in-the-loop, transparency, and dynamic knowledge – have ramifications beyond biology or even the scientific domain:

User-Guided Knowledge Incorporation

By elevating the user as a curator of data and process, the system effectively shifts accountability and fosters trust, a model that could be extended to healthcare decision-making, legal research, and any field requiring rigorously documented outputs. The user is collaborator and partner in the system by design.

Collaborative Intelligence

Instead of viewing AI as an autonomous oracle, the system reimagines AI as a collaborative partner. This has implications for interdisciplinary research teams, where multiple experts may iteratively refine a question. There is the potential for LmRaC to expand to a team setting where there is a mix of agents, human and LLM, working on a problem. Importantly, agents can be distributed.

Adaptation to Other Disciplines

The same modular approach can be deployed for finance, engineering, or social sciences, provided domain-specific knowledge bases, functions and APIs where available. LmRaC is focused on the biological sciences because of its author's interest, not due to any limitation in design.

8.6 Ethical and Social Implications

8.6.1 Ethical Dilemmas and Risks

User Privacy vs. Functionality

Scientific research often involves proprietary or sensitive datasets. The retrieval-augmented design means private data is uploaded and stored in vector databases. If managed correctly, this offers a layer of security, however, interface with external APIs must be managed with privacy in mind. This includes GPT that is behind many tools.

How You Define Hallucination

Although the RAG architecture greatly reduces spurious content, an LLM might still generate plausible but incorrect statements if the user inadvertently injects misinformation into the experiment index. Like all scientific inquiry, conclusions are only as good as the data. LmRaC attempts to balance reliability and trust with a user-centric design. This means that while users may now query their own data with LLM-based tools, it is incumbent on users to ensure the quality and veracity of this information.

8.6.2 Mitigations

Transparency

By exposing all references as paragraph-level links, users can immediately trace and verify the credibility of any statement.

Security and Access Control

Future versions can implement secure encryption and fine-grained access control for institutional or clinical environments. Also, because tools are wrapped in notes they can include auditable information regarding use and privacy.

8.7 Comparison to Existing Work

8.7.1 Advancing the State of the Art

Compared to single-shot or purely generative LLM systems, LmRaC introduces a genuine user-in-the-loop design. Most advanced LLM-based solution – whether commercial or open-source – lack built-in mechanisms to incorporate evolving user data while simultaneously ensuring fully traceable references. Emphasis is on improving human perform through the use of LLM-enabled tools, not in creating a better blackbox.

8.7.2 Key Distinctions

User-in-the-loop vs. Blackbox

LmRaC is deliberately designed around a collaborative notion of AI, while advanced autonomous models like OpenAI’s o1 and o3 are more akin to closed systems in that the user is not privy to the details of execution.

RAGs and Indexing

Traditional RAG-based systems often tie retrieval to a static knowledge base or a single embedding store. LmRaC, by using a multi-RAG, multi-index approach that divides domain and experiment knowledge into distinct bins enables a more granular, flexible and evolving architecture.

8.7.3 Limitations of Comparison

Direct comparisons to advanced closed-source models are difficult, as LmRaC prioritizes collaboration with the user. While those models may excel in speed or broad knowledge coverage, LmRaC’s advantage is in its traceable, domain-focused methodology and ability to evolve and adapt with the user. A more appropriate comparison, though

beyond the scope of this current research would be a controlled comparison between users working either with or without the collaboration of LmRaC. This might entail tasks involving data analysis and discovery such as those found in current benchmarks. For example, the GPQA [210] benchmark required expert human review. What if, testing is less about answering the question, than using LmRaC to confirm expert validation of the questions. The distinction is subtle, but important when considering human-in-the-loop systems like LmRaC.

CHAPTER 9 CONCLUSION

9.1 Innovation

The approach taken in this research represents a major step forward in the use LLMs as language reasoning systems. Traditionally, LLMs have been leveraged primarily for text generation and natural language understanding. However, LmRaC pushes this paradigm further by integrating a pre-screening RAG mechanism for "usefulness," which is crucial in domains where there is low dynamic range in embeddings, ensuring that only the most relevant and high-utility information is used. This innovation not only optimizes the quality of data fed into the LLM but also enhances the overall decision-making process by filtering out noise and irrelevant information and all but eliminating hallucinations. It also adds transparency to the origin of data sources.

Another innovation is the seamless integration of LLMs with human notetaking and reasoning processes. This dynamic interaction allows for a continuous and iterative exchange between human inputs and machine outputs, facilitating a more natural and productive workflow. Moreover, the design is extensible, allowing the system to flexibly accumulate experiential knowledge over time. This adaptability is essential for evolving scientific and research needs, where the ability to incorporate new findings and adjust reasoning processes is critical.

Importantly, the system incorporates quantitative data as a first-class type in LLM conversations. Unlike traditional systems where numerical data might be an afterthought or handled separately, the system treats user experimental data as integral to the reasoning process. This integration allows the system to perform sophisticated data-driven

analyses within the same framework that handles language processing, resulting in more comprehensive and accurate outputs especially in scientific research.

9.2 Practical Significance

This work significantly advances the state of the art in applications built around LLMs. While much of the current research focuses on fine-tuning LLMs for specific tasks, chat interfaces, or RAG-augmented chat systems, LmRaC introduces a truly integrative approach where the LLM becomes an integral component of a larger software system designed for complex, real-world applications. This approach shifts the role of LLMs from being mere tools for generating text to becoming central components of software architecture.

This work addresses several widely recognized weaknesses and limitations of LLM adoption in scientific research. This work treats the LLM as a reasoning engine and integrates it deeply into the human workflow, creating a collaborative framework where the LLM and human user work in tandem. This design ensures that the system is not a passive tool but an active participant in the reasoning process, enhancing both the accuracy and efficiency of scientific discovery.

9.3 Interdisciplinary Approach

The design of this system is interdisciplinary by nature, incorporating psychological principles to improve LLM performance. For instance, the system is designed to support cognitive workflows, ensuring that the interaction between the human user and the LLM is as intuitive and supportive as possible. This is particularly important in applications aimed at aiding scientific research, where cognitive load and the need for clear, logical reasoning

are paramount.

In terms of application, this work is designed to aid the scientific enterprise, with a particular focus on biological research. However, the design principles and methodologies are generalizable to other scientific domains, making it a potentially versatile tool for a wide range of research activities.

9.4 Ethical Computing

The power of LLMs raises significant ethical concerns, particularly regarding their use in critical domains like healthcare and scientific research. One of the primary issues is that LLMs, when used as black-box components, remain largely "unaccountable" with respect to many steps of reasoning. The approach taken here addresses this concern by shifting much of the critical reasoning to other components within the system, providing clear avenues for traceability and audit of the reasoning process.

Recent advances in LLM models (e.g., OpenAI o1 and o3) have continued to emphasize autonomy over transparency. No doubt this is driven in part by a business strategy that emphasize the value of proprietary methods over transparency. Trust simply becomes redefined as reliable with assurance that the latest model outperforms all other previous models and is smarter than the average PhD. We argue here that this approach is not only dangerous, but counterproductive. Humans *do* have valuable insights and experience to bring to a collaboration. The sum of world texts is much less than the measure of our potential.

Moreover, by adopting a collaborative approach this positions the system as a partner to the human user, rather than a replacement. This symbiotic relationship ensures that

while the LLM contributes to the reasoning process, final decisions remain in human hands, reducing the risks associated with fully autonomous AI systems. By integrating human oversight and interaction at every step, this work aims to create a more ethical and accountable system.

9.5 Foundation for Future Work

We expect this approach lays the foundation for future work by moving beyond static knowledge bases toward dynamic solution-based systems. In this framework, problem-solving is collaborative, goal-oriented, and continuously evolving. This dynamic approach is particularly suited to the practical aspects of scientific discovery, where new information and insights must be rapidly incorporated into the decision-making process.

By making the LLM a tool for more efficient and expansive human reasoning, this work offers a new paradigm for AI-human collaboration. The system is designed to enhance and extend human capabilities, providing a powerful platform for solving complex problems in science and beyond.

CHAPTER 10 FUTURE RESEARCH

This final section concludes with some ideas both considered and speculative for future extensions to the LmRaC framework.

10.1 Just in Cases

This section outlines planned enhancements to case-based reasoning (CBR) in order to improve its effectiveness and adaptability. The proposed improvements focus on multi-faceted similarity measures, weighted semantic graphs, mechanisms for discovery and synthesis, and interactive review and reflection components. These advancements aim to make CBR more dynamic, collaborative, and capable of learning from user interactions and past cases. By implementing these enhancements, we aim to make case-based reasoning systems more adaptable, insightful, and collaborative.

10.1.1 Multi-faceted Similarity

To enable more precise adaptation of cases, we plan to incorporate multi-faceted similarity measures. Traditional CBR systems often focus on holistic comparisons between cases, but this approach can miss opportunities for partial reuse or adaptation. The proposed measures will account not only for the overall objectives of a case but also for individual steps and data dependencies. This fine-grained similarity analysis will facilitate the flexible and fractional repurposing of cases, allowing users to adapt components of a case to novel problems.

For example, if a case involves solving a multi-step problem, the system could identify which steps are most relevant to a new situation while removing, revising or reinventing new steps to form a complete solution. This may also include asking the user for advice.

This capability is particularly important in complex domains where only portions of a prior solution may be applicable.

10.1.2 Weighted Semantic Graphs

Semantic graphs will serve as a critical tool for organizing and retrieving cases. Instead of being a rigid organizing structure, graphs can augment embeddings, providing cues that aid in search and retrieval. By assigning weights to relationships between nodes, we may also capture the strength of connections between concepts, enabling nuanced exploration of the case library.

For example, a weighted graph might indicate that two cases are loosely related based on overarching goals but share a strong connection in the data they work on or the tools that are employed. These soft semantic graphs will enhance the system's ability to make inferences, even when direct matches are unavailable, and enable more context-sensitive searches and exploration.

10.1.3 Discovery and Synthesis

CBR systems often struggle to support serendipitous discovery and synthesis across a large case library. To address this, we propose tools that incorporate random walks between notes and on their associated semantic graphs. Traditional search methods often focus on the shortest path between two nodes, but this approach can overlook less obvious but potentially valuable connections. Random walks introduce variability, allowing the system to uncover more distant or unconventional semantic associations between cases and user notes.

In addition to improving retrieval, these mechanisms will support adaptive learning by synthesizing insights from multiple sources. This synthesis capability will help identify

patterns and relationships that might otherwise remain hidden, enhancing the overall effectiveness of the case library.

10.1.4 Review and Reflection

A core element of the proposed enhancements is the integration of interactive and collaborative review mechanisms. These methods and tools will provide opportunities for the user and system to pause and reflect on a plan at various stages of its design and execution. Key features include:

- **Collaborative Plan Review:** After a plan is created, the system can prompt the user to review it before execution. Users can provide feedback, suggest modifications, or proceed "as-is." This feature ensures that plans align with user expectations and can be overridden based on preferences learned from past cases.
- **Step-by-Step Reflection:** The system can pause after each step in a plan to review outcomes and solicit user feedback. This iterative process allows for real-time adjustments and ensures greater accuracy in dynamic scenarios.
- **Post-Mortem Review:** Users and system will have the option to initiate reviews after a case is completed. These reviews will support evaluation of the adequacy of the approach and its outcomes, allowing for refinement based on new information.
- **Meta Review:** Mechanisms for higher-level reviews will enable tagging and annotating cases during or after execution. Tags can indicate the intended use of information – whether for answering specific questions, augmenting prompts, or enhancing future searches.

In all cases context will be applied such that some problems will be approached with careful consideration, while others may be more free to make speculative associations. At any point the user can indicate a change in approach.

10.2 Visualization as a Tool

A framework – "Second Thought" – to explore the use of visualization to enhance interactions with large language models (LLMs). Central to this concept is treating the LLM prompt as a state, where a succession of prompts then represents a flow. To enrich user control, the framework introduces mechanisms for rewinding to earlier states, editing prompts, and visualizing state changes.

A key feature of Second Thought is its visualization and interface design. States or prompts are conceptualized as documents that can be annotated with metadata, including user notes, critiques, and summaries. This metadata allows for more sophisticated meta-operations, such as *conditionality* – where an executive LLM (eLLM) interprets and modifies states based on the metadata – and *bifurcation*, enabling the creation of parallel or alternative threads. This effectively transforms the framework into a graphical programming language, where the LLM can manipulate state/prompt/documents dynamically.

Visualization tools like Obsidian or XMind could be integrated to display these documents as graphs (using Obsidian's built-in Graph View) or hierarchical maps, respectively. This design not only facilitates better collaboration by incorporating human input throughout the workflow but also introduces the eLLM as an active participant in decision-making. By enabling revisions, retries, and multi-threaded exploration, Second

Thought emphasizes iterative improvement and creativity in prompt design.

The framework's name, "Second Thought," reflects its purpose: empowering users to refine their interactions with LLMs furthering adaptability and aligning with the LmRaC vision of collaborative, flexible LLM utilization.

10.3 Agent Extensions

The following sections outline strategies to enhance and extend the capabilities of intelligent agents, focusing on agentic independence, the development of agentic personalities, and improvements in agentic attention and frustration handling.

10.3.1 Agentic Independence

A *Collaborator Agent* could shift focus from designing a monolithic executive system to creating an agent capable of independently navigating notes, managing tasks, and assisting in planning and execution. This agent would abstract the creation and execution of plans as a type of "case" or "recipe," enabling a structured approach to reasoning and task completion. This agent's reasoning processes could be built on foundational or cases like:

- **Assessment Cases:** Evaluate the current situation or problem.
- **Planning Case:** Formulate a course of action based on the assessment.
- **Reflection Case:** Analyze outcomes and refine approaches for future tasks.
- **Ponder Case:** Pause and consider broader implications or alternative strategies.

To support dynamic interactions, the agent could use a short-term memory stack. This stack would allow the agent to handle interruptions gracefully, maintaining state while

temporarily diverting attention to new tasks.

10.3.2 Agentic Personalities

To extend the adaptability of agents, the concept of *Agentic Personalities* introduces mechanisms for experiential learning, persistence of knowledge, and the evolution of capabilities over time. Features could include:

Persistence of Experience

Agents would maintain a "Note to Self" (NtS) system, recording experiences not as static facts but as reflections on problem-solving moments. These notes would serve as a form of episodic memory, embedding insights from past reasoning processes into future tasks. These NtS artifacts would be inspectable by humans, allowing for review, critique, and collaborative refinement.

Inheritance and Evolution

Agents would develop the ability to inherit foundational knowledge and adapt it over a lifetime. This approach mirrors biological evolution:

- **Inherit** - agents would start with a base of cultural knowledge recorded in artifacts, enabling transfer across generations.
- **Learn** - agents would incorporate new experiences and instructions into their operational frameworks.
- **Evolve** - over multiple iterations, agents would refine their capabilities and potentially develop new functionalities.

Collaborative Learning

Agents would support apprenticeship models, where humans can guide their learning processes. For instance, users might ask questions or demonstrate solutions, and agents could adapt their knowledge based on these interactions.

Cultural Knowledge

By preserving knowledge in artifacts, agents would maintain a form of "agentic culture" that transcends individual implementations. This shared knowledge would support new capabilities and provide a foundation for further evolution.

10.3.3 Agentic Attention, Frustration and Values

Agents might develop the ability to manage their focus and recognize when to change course, balancing perseverance with efficiency. This requires nuanced attention mechanisms that go beyond simple timers or counters associated with task effort. Agents would monitor their progress and recognize when a line of reasoning is yielding less valuable results. By assessing diminishing returns (i.e., frustration) they could decide to abandon unproductive paths in favor of alternative approaches. Over time, agents would build an "intuition" for fruitful approaches, distinguishing between critical tasks and peripheral distractions through a weighting measure. This capacity would enhance their ability to remain focused on objectives.

To guide their reasoning and self-evaluation, agents could maintain a long-term memory store encapsulating guiding "values." These values would help the agent: assess whether a goal has been met satisfactorily; align its actions with broader principles, such as efficiency, adaptability, or collaboration. Drawing inspiration from psychological

frameworks (e.g., transactional vs. relational values), these guiding principles would provide a basis for decision-making and refinement over time.

10.4 Creative Hallucination or Maybe Timothy Leary was Just Ahead of His Time

Although LLM hallucination is almost universally viewed as a limitation to be overcome, there is evidence that if leveraged for what it is, it can be a useful feature to be exploited. So, what *is* hallucination? In my opinion it stems from a misinterpretation of many users and commentators that it is a failure of the LLM to accurately and/or truthfully answer a question. Consider, however, that LLMs – or more accurately transformers – are not reasoning engines. They are prediction engines. They do not answer question; they return the most probable next series of tokens. It is easy to confuse an answer as the most probable series of tokens to follow a question, but that is not necessarily the case. And, when the question does not have a well established answer on which to train, what follows is simply a probable series of tokens. Hallucinations arise from this divergence between answers derived from reason and replies derived from probability.

This is an opportunity! Researchers like David Baker have turned this to an advantage and used hallucination as a means to predict what *could* be. In Baker's case he uses the technique to design de novo proteins [213, 214]. This idea of viewing hallucinations not as errors, but as plausible variants should be explored. Methods like these can be leveraged to generate creative connections and novel hypotheses. The hallucinate is essentially confirmation that something that isn't real near the boundary of the possible. Isn't this the definition of innovation and creativity?

Similarly, diffusion models (also used by Baker) [215–218] can create novel variants that can then be tested for the properties they are mimicking. I pose the question: could hallucination and diffusion be leveraged to enhance learning, problem solving, hypothesis generation/testing and scientific discovery?

APPENDIX A CONTROLLED VOCABULARIES

Genes, diseases and pathways are recognized in questions using controlled vocabulary files derived from standard sources (see below). Diseases and pathways are then converted into their standard accession number for purposes of metadata tagging. Genes simply use their canonical symbol (e.g., BRCA1) . This way when a literature search is performed on a gene, disease or pathway, associated chunks can be assigned one or more metadata values. These metadata values are then used in subsequent searches to improve results.

Genes

Human gene symbols (n=53,736) were downloaded from the Human Gene Organization (HUGO) along with the gene description, Entrez ID, ENSEMBL ID and UniProt ID [189]. Table 1 shows a sample of this vocabulary file.

symbol	name	entrez id	ensembl gene id	uniprot ids
A1BG	alpha-1-B glycoprotein	1	ENSG00000121410	P04217
A1BG-AS1	A1BG antisense RNA 1	503538	ENSG00000268895	
A1CF	APOBEC1 complementation factor	29974	ENSG00000148584	Q9NQ94
A2M	alpha-2-macroglobulin	2	ENSG00000175899	P01023
A2M-AS1	A2M antisense RNA 1	144571	ENSG00000245105	
A2ML1	alpha-2-macroglobulin like 1	144568	ENSG00000166535	A8K2U0
A2ML1-AS1	A2ML1 antisense RNA 1	100874108	ENSG00000256661	
A2ML1-AS2	A2ML1 antisense RNA 2	106478979	ENSG00000256904	
A2MP1	alpha-2-macroglobulin pseudogene 1	3	ENSG00000256069	
...

Table 1: Gene vocabulary file. Includes: gene symbol, name, Entrez identifier, Ensembl identifier and Uniprot protein identifier.

Diseases

Disease names and identifiers (n=5004) were downloaded from the NIH National Library of Medicine. Only identifiers from the "C" tree (i.e., diseases) were used [188]. Table 2 shows a sample of this vocabulary file.

MeSH ID	Disease
...	...
D000071956	Myocardial Contusions
D000071960	Breast Carcinoma In Situ
D000072039	Fracture Dislocation
D000072042	Salter-Harris Fractures
D000072281	Lymphadenopathy
D000072656	Breast Cancer Lymphedema
D000072657	ST Elevation Myocardial Infarction
D000072658	Non-ST Elevated Myocardial Infarction
...	...

Table 2: Disease vocabulary file. Includes: disease MeSH identifier and disease name.

Pathways

KEGG pathways (n=356) and their associated accession numbers were downloaded from KEGG [187]. Also included were all associated primary references used by KEGG for defining the pathway. These are stored as PubMed PMID identifiers. Table 3 shows a sample of this vocabulary file.

KEGG ID	Pathway Name	Primary References
...
hsa05210	Colorectal cancer	[15000146,16555243,11078609,16699851,12621137,12951588,10505543,...]
hsa05211	Renal cell carcinoma	[16339096,15122209,14634372,15611513,17211469,17287871,14685170,...]
hsa05212	Pancreatic cancer	[12459728,11407945,12946833,16702400,10740269,12648469,14967450,...]
hsa05213	Endometrial cancer	[15947972,14747944,16508724,17028294,11431710,15284264,7585656,...]
hsa05214	Glioma	[15639402,12154354,11253051,16700615,12762887,16095998,11057895,...]
hsa05215	Prostate cancer	[12878745,15724144,16551847,11900250,15082523,9422516,10835690,...]
hsa05216	Thyroid cancer	[16557281,16551846,16946003,17062879,16242838,16189702,11165748,...]
hsa05217	Basal cell carcinoma	[15948971,12588368,15978322,15468170,16881963,16306523,11276000,...]
hsa05218	Melanoma	[16822996,12894244,16750612,15841168,16001050,16001072,16899407,...]
hsa05219	Bladder cancer	[17158541,16474624,16110317,11103787,12850530,17149381,15746962,...]
...

Table 3: Pathway vocabulary file. Includes: pathway identifier (hsa=human species), pathway name and primary pathway references (PubMed PMID identifiers).

APPENDIX B TEST QUESTIONS

Table 4 lists questions used for evaluation of LmRaC and other LLM models, especially in the pairwise evaluation of LLM model performance (see Figure 11).

Complexity	Question
1	What is GNAI1 also known as?
2	What genes in the chemokine signaling pathway are significant with respect to sepsis?
2	What genes interact with INS in diabetes mellitus?
2	What genes are most studied in atherosclerosis and the lipid and atherosclerosis pathway?
2	What genes are deregulated in asthma?
3	What role does GNAI1 play in the chemokine signaling pathway?
3	What is the role of GNAI1 in colorectal cancer?
3	What role does TP53 play in colorectal cancer?
3	How is BRCA1 related to colorectal cancer?
3	How is Sepsis related to the chemokine signaling pathway?
3	How is sepsis related to pyrimidine metabolism?
3	How is mismatch repair related to colorectal cancer?
3	How is RAD23B related to the polypyrimidine tract and microsatellite instability?
3	What do sepsis and asthma have in common?
4	Is there evidence of any interaction between the genes APP and HTT in Alzheimer Disease?
4	What biological pathways and deregulation are common to Breast Cancer, Colorectal Cancer and Endometrial Cancer?
4	How does pediatric asthma differ from adult asthma?
4	How do BRCA1 and TP53 interact in cancer?
5	How might down regulation of both GNAI1 and ADCY5 be related to colon or colorectal (CRC) cancer? How does this relate to chemokine signaling?
5	How might up regulation of PIK3R2 along with down regulation of PIK3R5 affect the chemokine signaling pathway with respect to colon or colorectal cancer?

Table 4: Questions for testing and pairwise performance comparison between LLM models.

NOTE: Relative complexity is ranked 1 (easiest) to 5 (most difficult).

APPENDIX C SAMPLE COMPARISON RESULTS

An example of answers from each of the models (LmRaC, GPT4, GPT3, BioGPT, Llama2, PaLM2) for one of the comparison questions (Appendix B) along with pairwise comparison results for all answers to LmRaC. Note that for "Comparison: X to Y" references to "answer 1" refer to the answer from model X, while "answer 2" refers to the answer from model Y.

Question

How might up regulation of PIK3R2 along with down regulation of PIK3R5 affect the chemokine signaling pathway with respect to colon or colorectal cancer?

Answer: LmRaC

The role of PIK3R2 in chemokine signaling pathways, while not explicitly detailed in the provided citations, can be inferred from its general functions as a regulatory subunit of PI3K. PI3Ks, including those with the PIK3R2 subunit, are key signaling molecules activated by chemokine receptor-mediated signaling ([El Haibi, CP. et al., 2010](#)). These receptors activate Class IA PI3Ks, such as those with the p85 β regulatory subunit encoded by PIK3R2, leading to the conversion of PIP2 to PIP3 and the activation of downstream pathways like AKT, which promote cell survival, proliferation, and migration ([Liu, Y. et al., 2022](#))([Wang, Y. et al., 2021](#))([Vallejo-Díaz, J. et al., 2016](#)).

Given PIK3R2's role in encoding the p85 β regulatory subunit and the universal activation of PI3K by chemokine receptors ([Wang, Y. et al., 2021](#))([Thelen, F. et al., 2021](#)), PIK3R2 likely regulates PI3K activity in response to chemokine signaling. This

involves mediating the association of PI3K's catalytic subunit with activated receptors, influencing downstream events that regulate cellular responses ([Liu, Y. et al., 2022](#)). Although direct descriptions of PIK3R2's role are absent in the citations, its general involvement in PI3K pathways and specific functions suggest a contributory role in regulating chemokine-induced PI3K activity and subsequent cellular responses.

PIK3R2 upregulation affects colon or colorectal cancer (CRC) progression by activating the PI3K/AKT signaling pathway, which controls cell proliferation and apoptosis ([Zhou, X. et al., 2023](#)). Frequently overexpressed in cancers, including colon cancer, PIK3R2's increased expression leads to higher PIP3 levels and AKT signaling activation, potentially regulating malignancy progression ([Xu, HF. et al., 2019](#)) ([Zhou, X. et al., 2023](#)). In CRC, the PI3K/AKT pathway is crucial for regulating biological processes like cell proliferation and survival ([Nunez, SK. et al., 2021](#)). PIK3R2 dysregulation contributes to CRC progression by promoting uncontrolled proliferation and survival signals. Moreover, PIK3R2 overexpression can counteract the effects of YWHAB knockdown, which inhibits proliferation and induces apoptosis in colon cancer cells ([Zhou, X. et al., 2023](#)).

PIK3R2 overexpression is also associated with clinical prognosis, DNA methylation, and immune infiltration in various tumors, aiding in understanding its role in tumorigenesis ([Liu, Y. et al., 2022](#)). In CRC, PIK3R2 overexpression, along with mutations like KRAS and PIK3CA, can further activate the PI3K/AKT/mTOR pathway, important for predicting responses to targeted cancer therapies ([Oikonomou, E. et al., 2014](#)). Thus, PIK3R2 upregulation in CRC enhances the PI3K/AKT signaling pathway, regulating critical processes for tumor growth and metastasis.

Regarding interactions between PIK3R2 and PIK3R5 in cancer, the provided citations

do not mention any direct connections. While citation (Liu, Y. et al., 2022) discusses PIK3R1 and PIK3R2 in head and neck cancer and citation (Du, Z. et al., 2017) describes PIK3R5's broad protein interactions and post-translational modifications, neither specifies interactions with PIK3R2 in cancer. Citation (Lin, J. et al., 2021) focuses on PIK3R2 in liver cancer, interacting with EFNA4 and EPHA2 but not PIK3R5. Similarly, citation (Du, J. et al., 2020) mentions PIK3R5 overexpression in cancer stem cells and its role in the PI3K/AKT pathway without detailing interactions with PIK3R2. Citation (Liu, Y. et al., 2022) does not provide additional information on the interaction between the two. Therefore, the known interactions between PIK3R2 and PIK3R5 in the context of cancer remain unspecified.

REFERENCES

- El Haibi, CP; Sharma, PK; Singh, R; Johnson, PR; Suttles, J; Singh, S; Lillard, JW. PI3Kp110-, Src-, FAK-dependent and DOCK2-independent migration and invasion of CXCL13-stimulated prostate cancer cells. *Molecular cancer*. 2010. [PMC2873439](#)
- Liu, Y; Wang, D; Li, Z; Li, X; Jin, M; Jia, N; Cui, X; Hu, G; Tang, T; Yu, Q. Pan-cancer analysis on the role of PIK3R1 and PIK3R2 in human tumors. *Scientific reports*. 2022. [PMC8993854](#)
- Nunez, SK; Young, CD; Griffen, TL; Ohandjo, AQ; McKinney, LP; Kopetz, S; Lillard, JW. Identification of Gene Co-Expression Networks Associated with Consensus Molecular Subtype-1 of Colorectal Cancer. *Cancers*. 2021. [PMC8616344](#)
- Oikonomou, E; Koustas, E; Goulielmaki, M; Pintzas, A. BRAF vs RAS oncogenes: are mutations of the same pathway equal? Differential signalling and therapeutic implications. *Oncotarget*. 2014. [PMC4322985](#)
- Thelen, F; Wissmann, S; Ruef, N; Stein, JV. The Tec Kinase Itk Integrates Naïve T Cell

Migration and In Vivo Homeostasis. *Frontiers in immunology*. 2021. [PMC8458560](#)

Vallejo-Díaz, J; Olazabal-Morán, M; Cariaga-Martínez, AE; Pajares, MJ; Flores, JM; Pio, R; Montuenga, LM; Carrera, AC. Targeted depletion of PIK3R2 induces regression of lung squamous cell carcinoma. *Oncotarget*. 2016. [PMC5356720](#)

Wang, Y; Peng, J; Bai, S; Yu, H; He, H; Fan, C; Hao, Y; Guan, Y. A PIK3R2 Mutation in Familial Temporal Lobe Epilepsy as a Possible Pathogenic Variant. *Frontiers in genetics*. 2021. [PMC8141861](#)

Xu, HF; Huang, TJ; Yang, Q; Xu, L; Lin, F; Lang, YH; Hu, H; Peng, LX; Meng, DF; Xie, YJ; Tan, L; Qian, CN; Huang, BJ. Candidate tumor suppressor gene IRF6 is involved in human breast cancer pathogenesis via modulating PI3K-regulatory subunit PIK3R2 expression. *Cancer management and research*. 2019. [PMC6594015](#)

Zhou, X; Chen, A; Zhang, T. YWHAB knockdown inhibits cell proliferation whilst promoting cell cycle arrest and apoptosis in colon cancer cells through PIK3R2. *Experimental and therapeutic medicine*. 2023. [PMC10119628](#)

Answer: BioGPT

How might up regulation of PIK3R2 along with down regulation of PIK3R5 affect the chemokine signaling pathway with respect to colon or colorectal cancer? PIK3R2 is the regulatory subunit of PI3K that negatively regulates the catalytic subunit, PIK3CA. Thus, it was proposed that a decrease in PIK3R5 levels relieves the inhibition of PI3K, PIK3R2 and PIK3CA collectively resulting in increased PIP3 and Akt levels, which is known to be a frequent event in cancers. In addition, overexpression of PIK3R2 is often correlated with resistance to inhibitors of the PI3K α pathway (6, 7), while loss of the p8 5 α regulatory

subunit of PI3K can also result in constitutive activation of the kinase. PIK3R2 has also been shown to regulate angiogenesis and cell survival. PIK3R5 forms heterodimers with both class IA PI3K subunits p110 α and p110 β , resulting in a decreased lipid kinase activity of PI3K α in vitro compared to that of PI3K α alone. However, it has also been reported that PIK3R5 can function as a positive regulator of PI3K signaling. PIK3R5-deficient mouse embryonic fibroblasts (MEFs) show decreased Akt phosphorylation and are resistant to oncogenic transformation by constitutively active Akt. In addition, we previously showed that knock down of PIK3R5 in HT-29 cells results in decreased cellular PIP3 levels and decreased activation of Akt. PIK3R2 shows higher expression levels in colon compared to rectum and rectum compared to other tissues (Figure 1). Furthermore, in colon cancer there is a correlation between high expression levels of PIK3R2 and poor prognosis in patients with colon cancer (p Stage I & II: p = 0.0095, p Stage III & IV: p = 0.03) (Figure 2A). In addition, patients with high expression of both PIK3R2 and PIK3CA have a significantly worse prognosis compared to patients with low expression of both genes (p = 0.002, HR: 2.49). This observation supports the findings in the previous section that, in colon cancer, high expression of PIK3R2 is associated with poor prognosis, and suggests that this poor prognosis might at least in part be due to increased PIK3CA activity. Although, the mechanism for the positive effect of PIK3R2 on prognosis in colon cancer still needs to be elucidated, we recently found that knock down of PIK3R2 results in decreased cellular levels of PIP3 and decreased activation of Akt, suggesting that PIK3R2 could exert its effect also through its kinase-independent scaffolding function. Interestingly, high expression of PIK3CA has been correlated with good prognosis in colon cancer (2, 5). Based on these findings, it is tempting to speculate

that high expression of PIK3R2 might increase PIP3 and Akt levels at earlier stages in tumor development. Increased PIP3 and Akt activity might affect apoptosis induction and other signaling pathways in a way that results in faster tumor progression. Another possible explanation for the correlation between high PIK3R2 levels and poor prognosis in colon cancer could be related to an effect of PIK3R2 on cell motility. In this context it is interesting to note that PIK3R5 has been shown to affect cell motility and that knock down of PIK3R5 resulted in reduced chemotaxis in breast cancer cells and MEFs. Furthermore, it was also shown that the effects of PIK3R5 on cell migration were dependent on its localization at the leading edge of the cells (10, 11). Knock down of PIK3R5 resulted in reduced membrane ruffling and decreased activation of Rac1 and Cdc42. Moreover, it was suggested that PIK3R5 functions by forming heterodimers with PI3K regulatory subunits (mainly PIK3R1) and positively regulates their lipid kinase activity, since it could be demonstrated that knock down of PIK3R5 results in decreased levels of PIP3 and PtdIns (3, 4) P2. In addition, it was shown that p110 β , PIK3R1, and PIK3R5 could be purified from brain by immunoprecipitation with an antibody against PIK3R5, and this complex was able to phosphorylate PtdIns (4, 5) P2 in vitro. Based on these findings it was proposed that a large proportion of the cellular PI3K activity could be channeled through the p85 β -p110 β heterodimeric complex, suggesting that the PI3K pathway could be predominantly regulated via the p85 β regulatory subunit. Thus, one could speculate that an increased PIK3R2 / PIK3R5 expression ratio results in increased PI3K activity that affects cellular migration and survival, contributing to the poor prognosis in patients with high PIK3R2 expression. This hypothesis is supported by our previous findings that knock down of PIK3R5 results in decreased PIP3 and Akt levels and increased apoptosis induction in colon

cancer cells. Knock down of PIK3R5 resulted in decreased invasion of colon cancer cells in vitro. Furthermore, knock down of PIK3R5 inhibited the growth of tumors formed by HT-29 cells in mice, indicating that PIK3R5 has a function in tumor growth in vivo. In agreement

Answer: GPT3

The PI3K/Akt signaling pathway plays a crucial role in regulating various cellular processes, including cell growth, proliferation, survival, and migration. Dysregulation of this pathway has been implicated in the development and progression of various cancers, including colon or colorectal cancer.

PI3K (phosphoinositide 3-kinase) is a key enzyme in the PI3K/Akt pathway, and it is composed of a regulatory subunit (p85) and a catalytic subunit (p110). The regulatory subunit p85 is encoded by several genes, including PIK3R2 and PIK3R5. Upregulation of PIK3R2 and downregulation of PIK3R5 can have significant effects on the PI3K/Akt pathway and subsequently impact the chemokine signaling pathway in the context of colon or colorectal cancer.

In the context of colon or colorectal cancer, dysregulation of the chemokine signaling pathway can influence various aspects of tumor progression, including tumor cell proliferation, invasion, and metastasis. Chemokines are small signaling proteins that regulate the migration and activation of immune cells, as well as tumor cells, within the tumor microenvironment.

Upregulation of PIK3R2, which encodes a regulatory subunit of PI3K, can lead to increased activation of the PI3K/Akt pathway. This, in turn, can promote cell survival,

proliferation, and migration, which are all hallmarks of cancer progression. Additionally, the upregulation of PIK3R2 may lead to increased activation of downstream targets of the PI3K/Akt pathway, such as mTOR (mammalian target of rapamycin), which further promotes cell growth and survival.

Conversely, downregulation of PIK3R5 may lead to decreased inhibition of the PI3K/Akt pathway, as the p85 regulatory subunit encoded by PIK3R5 normally acts to suppress PI3K activity. This can result in increased PI3K/Akt signaling and its downstream effects on cell proliferation, survival, and migration.

The dysregulation of PIK3R2 and PIK3R5 can also impact the chemokine signaling pathway. Chemokines exert their effects through binding to specific G protein-coupled receptors (GPCRs) on the cell surface, leading to the activation of downstream signaling pathways, including the PI3K/Akt pathway. Dysregulation of PIK3R2 and PIK3R5 can alter the sensitivity of tumor cells to chemokine signaling, leading to enhanced migration, invasion, and metastasis.

In summary, upregulation of PIK3R2 and downregulation of PIK3R5 can have significant effects on the PI3K/Akt pathway and subsequently impact the chemokine signaling pathway in the context of colon or colorectal cancer. These molecular changes can contribute to the dysregulation of cellular processes involved in cancer progression, including cell proliferation, survival, and migration. Understanding the interplay between these signaling pathways may provide insights into potential therapeutic targets for the treatment of colon or colorectal cancer.

Answer: GPT4

To understand how the upregulation of PIK3R2 and downregulation of PIK3R5 might affect the chemokine signaling pathway in the context of colon or colorectal cancer, it is essential to first understand the roles of these genes and the pathways they are involved in.

PIK3R2 and PIK3R5 encode for regulatory subunits of phosphoinositide 3-kinases (PI3Ks), which are critical enzymes involved in cellular functions such as growth, proliferation, differentiation, motility, survival, and intracellular trafficking. PI3Ks are activated by various cell-surface receptors, including chemokine receptors, and are involved in the transduction of signals from these receptors to downstream effectors.

PIK3R2 encodes for the p85 β regulatory subunit, while PIK3R5 encodes for the p101 regulatory subunit, which are part of class IA and class IB PI3Ks, respectively. The regulatory subunits are responsible for stabilizing the p110 catalytic subunit and for mediating interactions with activated receptors and other adaptor proteins.

Chemokines are small cytokines that play a crucial role in the immune response by directing the migration of immune cells to sites of inflammation, infection, or injury. Chemokine signaling is also implicated in cancer progression and metastasis. Chemokine receptors, which are G protein-coupled receptors (GPCRs), can activate the PI3K/AKT pathway, leading to various cellular responses.

In the context of colon or colorectal cancer, the PI3K/AKT pathway is often dysregulated, and its activation is associated with tumorigenesis, tumor growth, survival, angiogenesis, and metastasis. Upregulation of PIK3R2 (p85 β) could potentially lead

to increased activation of the PI3K/AKT pathway. This could enhance the response to chemokines and amplify the signals that promote cell survival, proliferation, and migration, which are detrimental in the context of cancer as they could contribute to tumor growth and metastasis.

Conversely, downregulation of PIK3R5 (p101) might have a more complex effect. Since p101 is associated with class IB PI3Ks, which are primarily activated by GPCRs, its downregulation could lead to reduced PI3K signaling in response to chemokines. This might result in a decreased migratory response of immune cells to the tumor microenvironment, potentially affecting the immune surveillance against the tumor. However, it could also impair the tumor cells' own response to chemokine signaling, which might reduce their invasive and metastatic potential.

The net effect of these changes on the chemokine signaling pathway and colorectal cancer progression would depend on the balance between these opposing actions and the specific cellular context. It is also important to consider that the PI3K/AKT pathway is subject to regulation by multiple factors, and the impact of changes in PIK3R2 and PIK3R5 expression levels might be modulated by other signaling events and interactions within the tumor microenvironment.

In summary, the upregulation of PIK3R2 and downregulation of PIK3R5 could have significant but complex effects on the chemokine signaling pathway in colon or colorectal cancer. These changes could potentially enhance tumor progression by promoting survival and proliferation signals while also affecting the immune response to the tumor. Understanding these effects in detail requires further experimental investigation, including the use of *in vitro* and *in vivo* models of colorectal cancer.

Answer: Llama2

Based on my knowledge as an expert researcher in biology, I can provide a detailed explanation of how upregulation of PIK3R2 and downregulation of PIK3R5 may impact the chemokine signaling pathway in the context of colon or colorectal cancer.

PIK3R2 and PIK3R5 are both phosphatidylinositol-4,5-bisphosphate 3-kinase (PI3K) regulatory subunits that play critical roles in various cellular processes, including cell survival, proliferation, and migration. In the context of colon or colorectal cancer, altered expression of these proteins has been observed, and their dysregulation can contribute to tumorigenesis and disease progression.

Upregulation of PIK3R2 has been associated with enhanced activation of the PI3K/Akt signaling pathway, which is known to promote cell survival and proliferation. This could potentially contribute to the development and growth of colon or colorectal tumors. Additionally, PIK3R2 has been shown to interact with other key players in the chemokine signaling pathway, such as CXCL12 and its receptor CXCR7, to modulate their activity. Therefore, upregulation of PIK3R2 could also influence the recruitment and retention of immune cells within the tumor microenvironment, potentially affecting the efficacy of immunotherapies targeting these cells.

Downregulation of PIK3R5, on the other hand, may have opposing effects on the chemokine signaling pathway. PIK3R5 has been implicated in the regulation of CXCL12 production by cancer cells, and its loss may lead to decreased levels of this chemokine. CXCL12 is a potent attractant for immune cells, including T cells and macrophages, and its reduction may limit the ability of these cells to migrate towards the tumor site. Moreover,

PIK3R5 has been shown to interact with the EGFRvIII protein, which is overexpressed in many colon cancers, and regulates its activity. Therefore, downregulation of PIK3R5 may also impact the activity of EGFRvIII and influence the behavior of colon cancer cells.

In summary, altered expression of PIK3R2 and PIK3R5 in colon or colorectal cancer may have significant consequences for the chemokine signaling pathway, which plays a critical role in shaping the tumor microenvironment and influencing the effectiveness of immunotherapies. Upregulation of PIK3R2 could enhance pro-survival signals and promote cell growth, while downregulation of PIK3R5 might reduce the attraction of immune cells to the tumor site, potentially hindering the efficacy of immunotherapeutic approaches. Further research is needed to fully elucidate the mechanisms underlying these interactions and to identify potential therapeutic targets for colon or colorectal cancer treatment.

Answer: PaLM2

Upregulation of PIK3R2 and downregulation of PIK3R5 could affect the chemokine signaling pathway in colon or colorectal cancer by altering the expression of chemokines and their receptors. This could lead to changes in the migration and invasion of cancer cells, as well as the immune response to the cancer.

PIK3R2 is a member of the phosphoinositide 3-kinase (PI3K) family of proteins, which are involved in a variety of cellular processes, including cell growth, differentiation, and survival. PIK3R2 is specifically involved in the regulation of cell migration and invasion. PIK3R5 is another member of the PI3K family, but it has been shown to inhibit the activity of PIK3R2. This suggests that PIK3R5 may play a role in suppressing the migration and

invasion of cancer cells.

In colon or colorectal cancer, the expression of PIK3R2 is often increased, while the expression of PIK3R5 is decreased. This could lead to an increase in the migration and invasion of cancer cells, as well as a decrease in the immune response to the cancer.

The chemokine signaling pathway is a complex network of interactions between chemokines and their receptors. Chemokines are small proteins that are secreted by cells and act as signals to attract or repel other cells. Chemokine receptors are proteins on the surface of cells that bind to chemokines and trigger a cascade of intracellular signaling events.

In colon or colorectal cancer, the expression of chemokines and their receptors is often altered. This could lead to changes in the migration and invasion of cancer cells, as well as the immune response to the cancer.

Upregulation of PIK3R2 and downregulation of PIK3R5 could affect the chemokine signaling pathway in colon or colorectal cancer by altering the expression of chemokines and their receptors. This could lead to changes in the migration and invasion of cancer cells, as well as the immune response to the cancer. These changes could contribute to the development and progression of colon or colorectal cancer.

Comparison: BioGPT to LmRaC

"**answer1_quality**":

"Answer #1 provides a detailed explanation of the roles of PIK3R2 and PIK3R5 in the chemokine signaling pathway and their potential effects on colon or colorectal cancer. The answer includes specific mechanisms by which these regulatory subunits may influence

PI3K activity, PIP3 and Akt levels, and cellular processes such as angiogenesis, cell survival, and motility. It also discusses the implications of PIK3R2 and PIK3R5 expression levels on cancer prognosis and tumor growth, supported by references to experimental findings. However, the answer could be improved by providing direct citations to support the claims made."

"answer2_quality":

"Answer #2 provides a structured response with clear headings and references to support the statements made. The answer focuses on the general functions of PIK3R2 as a regulatory subunit of PI3K and its role in the chemokine signaling pathway. It also discusses the implications of PIK3R2 upregulation on colon or colorectal cancer progression and prognosis. However, the answer does not address the specific question of how the upregulation of PIK3R2 along with the downregulation of PIK3R5 would affect the chemokine signaling pathway, and it lacks detail on the role of PIK3R5."

"answers_comparison":

"Both answers discuss the role of PIK3R2 in the chemokine signaling pathway and its implications for colon or colorectal cancer. They agree on the importance of PIK3R2 in regulating PI3K activity and the downstream effects on cell proliferation, survival, and motility. However, they differ in their approach and depth of information. Answer #1 provides a more comprehensive explanation of the roles of both PIK3R2 and PIK3R5, while Answer #2 focuses primarily on PIK3R2 and does not address the combined effect of PIK3R2 upregulation and PIK3R5 downregulation. Answer #1 also discusses the prognostic implications of PIK3R2 expression levels in more detail than Answer index2."

"answer1_score": 7

"answer2_score": 5

Comparison: GPT3 to LmRaC

"answer1_quality":

"Answer #1 provides a comprehensive explanation of how upregulation of PIK3R2 and downregulation of PIK3R5 could affect the chemokine signaling pathway in the context of colon or colorectal cancer. The answer correctly identifies the role of PI3K/Akt signaling in cancer progression and the potential consequences of dysregulation in the regulatory subunits encoded by PIK3R2 and PIK3R5. The answer is well-structured, uses appropriate scientific terminology, and logically connects the molecular changes to potential impacts on cancer progression. However, it lacks specific references to scientific literature, which would strengthen the credibility of the information provided."

"answer2_quality":

"Answer #2 is detailed and includes specific references to scientific literature, which adds credibility to the information presented. The answer focuses on the role of PIK3R2 in chemokine signaling and its implications for colon or colorectal cancer progression. It also discusses the potential effects of PIK3R2 upregulation on the PI3K/AKT signaling pathway. However, the answer does not directly address the downregulation of PIK3R5 and its combined effect with PIK3R2 upregulation on the chemokine signaling pathway. Additionally, the answer includes a section labeled 'REFERENCES' with citations, which is beneficial for an expert reader seeking to verify the information or explore the topic further."

"answers_comparison":

"Both answers agree on the importance of the PI3K/Akt signaling pathway in cancer progression and the role of PIK3R2 in this process. They both suggest that upregulation of PIK3R2 can lead to increased activation of the PI3K/Akt pathway, promoting cancer cell survival, proliferation, and migration. However, Answer index1.discusses the combined effect of PIK3R2 upregulation and PIK3R5 downregulation on the chemokine signaling pathway, while Answer #2 focuses primarily on PIK3R2 and does not elaborate on the role of PIK3R5. Answer #2 provides a more in-depth analysis with references to scientific literature, which is missing in Answer index1. Conversely, Answer #1 provides a more balanced view of the potential effects of both PIK3R2 and PIK3R5 on the chemokine signaling pathway."

"answer1_score": 7**"answer2_score": 8**

Comparison: GPT4 to LmRaC**"answer1_quality":**

"Answer #1 provides a comprehensive and well-explained overview of the potential effects of PIK3R2 upregulation and PIK3R5 downregulation on the chemokine signaling pathway in the context of colon or colorectal cancer. The answer demonstrates a clear understanding of the roles of PIK3R2 and PIK3R5, the PI3K/AKT pathway, and the implications of their dysregulation in cancer. It also appropriately discusses the complexity of the effects and the need for further experimental investigation."

"answer2_quality":

"Answer #2 is less focused and contains a mix of relevant and irrelevant information. The answer includes references to scientific articles, which adds credibility, but it also includes a section on the role of PIK3R2 and PIK3R5 in cancer that does not directly address the question. The answer could be improved by providing a more direct analysis of the effects of PIK3R2 upregulation and PIK3R5 downregulation on the chemokine signaling pathway specifically in colon or colorectal cancer."

"answers_comparison":

"Both answers agree on the importance of PIK3R2 and PIK3R5 in the PI3K/AKT pathway and their potential impact on cancer progression. They both suggest that upregulation of PIK3R2 could lead to increased activation of the PI3K/AKT pathway, which is associated with tumor growth and metastasis. However, Answer #1 provides a more detailed and focused explanation of the potential effects on the chemokine signaling pathway, while Answer #2 includes a broader discussion of PIK3R2 and PIK3R5's roles in cancer without specifically tying it to the chemokine signaling pathway in colon or colorectal cancer. Answer #2 also includes a large number of references, which could be seen as both a strength and a weakness, as it may distract from the specific question at hand."

"answer1_score" : 8

"answer2_score" : 5

Comparison: Llama2 to LmRaC

"answer1_quality":

"Answer #1 provides a comprehensive and plausible explanation of how PIK3R2

upregulation and PIK3R5 downregulation might affect the chemokine signaling pathway in colon or colorectal cancer. The answer integrates knowledge of the PI3K/Akt signaling pathway, interactions with chemokine receptors, and potential effects on immune cell recruitment and tumor microenvironment. The answer is well-structured and demonstrates a clear understanding of the subject matter, making it highly credible and informative."

"answer2_quality":

"Answer #2 is detailed and includes references to support the claims made about the role of PIK3R2 in chemokine signaling pathways and its impact on colon or colorectal cancer. However, the answer is less focused on the specific question about the combined effect of PIK3R2 upregulation and PIK3R5 downregulation. Instead, it provides a general overview of PIK3R2's role in cancer and cites literature that does not directly address the question. The answer could be improved by directly addressing the combined effects of PIK3R2 and PIK3R5 on the chemokine signaling pathway."

"answers_comparison":

"Both answers agree on the role of PIK3R2 in activating the PI3K/Akt signaling pathway and its implications in cancer cell proliferation and survival. They also acknowledge the importance of this pathway in the context of colon or colorectal cancer. However, Answer #1 provides a more direct response to the question by discussing the potential combined effects of PIK3R2 upregulation and PIK3R5 downregulation on the chemokine signaling pathway. Answer

index2, while informative, does not specifically address the downregulation of PIK3R5 and its potential interactions with PIK3R2, which is a key aspect of the question."

"answer1_score": 9

"answer2_score": 7

Comparison: PaLM2 to LmRaC

"answer1_quality":

"Answer #1 provides a general overview of how upregulation of PIK3R2 and downregulation of PIK3R5 might affect the chemokine signaling pathway in colon or colorectal cancer. It correctly identifies PIK3R2 and PIK3R5 as members of the PI3K family and discusses their potential roles in cell migration, invasion, and immune response. However, the answer lacks specific details, references to scientific literature, and a deeper explanation of the molecular mechanisms involved."

"answer2_quality":

"Answer #2 provides a detailed and well-referenced explanation of the potential effects of PIK3R2 upregulation and PIK3R5 downregulation on the chemokine signaling pathway in colon or colorectal cancer. It includes specific information about the role of PIK3R2 in the PI3K/AKT signaling pathway and its implications for cancer progression. The answer also discusses the clinical relevance of PIK3R2 expression and its interaction with other genetic mutations. However, it does not directly address the downregulation of PIK3R5 and its potential interactions with PIK3R2."

"answers_comparison":

"Both answers agree on the importance of PIK3R2 in the PI3K/AKT signaling pathway and its potential impact on colon or colorectal cancer progression. They also both mention the role of PIK3R2 in cell migration and invasion. However, Answer #1 is more general

and lacks the depth and references provided in Answer index2. Answer #2 provides a more comprehensive and scientifically supported explanation but does not address the downregulation of PIK3R5 as explicitly as Answer #1 does. Neither answer provides a detailed discussion on the direct interaction between PIK3R2 and PIK3R5."

"answer1_score": 5

"answer2_score": 8

APPENDIX D EXAMPLE ANSWER

The following is a complete unedited question and answer, including LmRaC supplied citation links and references.

QUESTION: How might down regulation of both GNAI1 and ADCY5 be related to colon or colorectal (CRC) cancer? How does this relate to chemokine signaling?

ANSWER: The downregulation of GNAI1 and ADCY5 may be associated with colon or colorectal cancer (CRC) due to their roles in signal transduction pathways essential for normal cellular functions. GNAI1, part of the G α inhibitory family, transduces extracellular signals to downstream molecules, affecting proliferation, adhesion, and differentiation ([Yao, J. et al., 2012](#)). Although its specific role in CRC is not detailed in the citations, its involvement in cancer-related signaling pathways indicates that its downregulation could contribute to CRC progression.

ADCY5 is responsible for converting ATP to cyclic AMP (cAMP) and is implicated in tumor neovascularization and inflammatory infiltration ([Wei, J. et al., 2022](#)). While direct evidence of ADCY5's role in CRC is not provided, its participation in these processes may affect tumor growth and the tumor microenvironment, crucial to CRC development and progression.

Chemokines and cytokines facilitate CRC progression by enabling cell migration and metastasis, as evidenced by reduced CXCL12 signaling in CRC cells ([Goita, AA. et al., 2022](#)). The downregulation of GNAI1 could impact chemokine signaling pathways, as

chemokines bind to G-protein-coupled receptors involved in tumorigenesis, metastasis, and angiogenesis (Stefani, C. et al., 2021). However, without further information, the direct link between GNAI1, ADCY5, and chemokine signaling in CRC remains unclear.

ADCY5 has been identified in various tumors, including glioma, prostate, and lung cancer, and is involved in neovascularization and inflammatory infiltration (Wei, J. et al., 2022). In CRC, ADCY5 is one of six metabolism-related genes valuable in a prognostic model, predicting patient outcomes and associated with cancer progression and poor prognosis (Miao, Y. et al., 2020). Its methylation status and mRNA expression pattern are similar in patients with type 2 diabetes mellitus (T2DM) and CRC, suggesting a comorbid gene role (Wei, J. et al., 2022). Thus, ADCY5 may be significant in CRC pathogenesis and serve as a biomarker for prognosis (Miao, Y. et al., 2020).

Chemokine signaling plays a multifaceted role in CRC by mediating tumor pathogenesis, development, metastasis, and shaping the tumor microenvironment (TME) (Gao, SH. et al., 2021)(Mlecnik, B. et al., 2011)(Liu, Q. et al., 2016)(Zhang, XF. et al., 2023)(Belov, L. et al., 2010)(Zaravinos, A. et al., 2019)(Xu, L. et al., 2017)(Xu, J. et al., 2022)(Grazioso, TP. et al., 2019)(Stefani, C. et al., 2021)(Xu, J. et al., 2022)(Zhao, R. et al., 2022)(Miller, MC. et al., 2017)(Korbecki, J. et al., 2021)(Rizeq, B. et al., 2020)(Rivas-Fuentes, S. et al., 2015)(Rivas-Fuentes, S. et al., 2015)(Lai, WY. et al., 2021)(Liu, Y. et al., 2020)(Shi, Y. et al., 2020). The CXCL13/CXCR5 axis is involved in tumor cell metastasis and intestinal tumorigenesis (Gao, SH. et al., 2021). Chemokines recruit immune cells to the TME, influencing their function and contributing to immune suppression and tumorigenesis (Gao, SH. et al., 2021)(Zaravinos, A. et al., 2019). Receptors like CXCR3 and CXCR4 are linked to metastasis and poorer prognosis (Belov,

L. et al., 2010). Certain chemokines are associated with effective immune responses and favorable outcomes, while others do not contribute to an efficient immune contexture (Mlecnik, B. et al., 2011). CXCL8 signaling is related to angiogenesis and chemoresistance, correlating with tumor progression (Liu, Q. et al., 2016). Chemokines also recruit immune cells to the TME, affecting tumor progression and metastasis (Xu, J. et al., 2022). IL-6, highly expressed in CRC, controls cancer initiation, progression, and angiogenesis through signaling pathways like JAK/STAT3 and YAP (Grazioso, TP. et al., 2019)(Stefani, C. et al., 2021). Chemokines and their receptors are potential targets for new anti-tumor drugs (Xu, J. et al., 2022). CXCL12 directly stimulates signaling pathways promoting cancer cell growth, metastasis, and angiogenesis, and indirectly recruits cancer cells to CXCL12-expressing organs (Shi, Y. et al., 2020). In summary, chemokine signaling is crucial to CRC progression and regulation, impacting the TME, immune cell recruitment and function, tumor growth and metastasis, angiogenesis, and potential therapeutic targets.

REFERENCES:

- Belov, L; Zhou, J; Christopherson, RI. Cell surface markers in colorectal cancer prognosis. International journal of molecular sciences. 2010. [PMC3039945](#)
- Gao, SH; Liu, SZ; Wang, GZ; Zhou, GB. CXCL13 in Cancer and Other Diseases: Biological Functions, Clinical Significance, and Therapeutic Opportunities. Life (Basel, Switzerland). 2021. [PMC8708574](#)
- Gao, X; Liu, Q; Chen, X; Chen, S; Yang, J; Liu, Q; Cheng, Y. Screening of tumor grade-related mRNAs and lncRNAs for Esophagus Squamous Cell Carcinoma. Journal of clinical

laboratory analysis. 2021. [PMC8183932](#)

Goïta, AA; Guenot, D. Colorectal Cancer: The Contribution of CXCL12 and Its Receptors CXCR4 and CXCR7. Cancers. 2022. [PMC8997717](#)

Grazioso, TP; Brandt, M; Djouder, N. Diet, Microbiota, and Colorectal Cancer. iScience. 2019. [PMC6889474](#)

Korbecki, J; Bajdak-Rusinek, K; Kupnicka, P; Kapczuk, P; Simińska, D; Chlubek, D; Baranowska-Bosiacka, I. The Role of CXCL16 in the Pathogenesis of Cancer and Other Diseases. International journal of molecular sciences. 2021. [PMC8036711](#)

Lai, WY; Mueller, A. Latest update on chemokine receptors as therapeutic targets. Biochemical Society transactions. 2021. [PMC8286821](#)

Liu, Q; Li, A; Tian, Y; Wu, JD; Liu, Y; Li, T; Chen, Y; Han, X; Wu, K. The CXCL8-CXCR1/2 pathways in cancer. Cytokine & growth factor reviews. 2016. [PMC6142815](#)

Liu, Y; Li, C; Dong, L; Chen, X; Fan, R. Identification and verification of three key genes associated with survival and prognosis of COAD patients via integrated bioinformatics analysis. Bioscience reports. 2020. [PMC7522359](#)

Miao, Y; Li, Q; Wang, J; Quan, W; Li, C; Yang, Y; Mi, D. Prognostic implications of metabolism-associated gene signatures in colorectal cancer. PeerJ. 2020. [PMC7474523](#)

Miller, MC; Mayo, KH. Chemokines from a Structural Perspective. International journal of molecular sciences. 2017. [PMC5666770](#)

Mlecnik, B; Bindea, G; Pagès, F; Galon, J. Tumor immunosurveillance in human cancers. Cancer metastasis reviews. 2011. [PMC3044219](#)

Rivas-Fuentes, S; Salgado-Aguayo, A; Pertuz Bellosio, S; Gorocica Rosete, P; Alvarado-Vásquez, N; Aquino-Jarquin, G. Role of Chemokines in Non-Small Cell Lung Cancer:

Angiogenesis and Inflammation. Journal of Cancer. 2015. [PMC4543754](#)

Rizeq, B; Malki, MI. The Role of CCL21/CCR7 Chemokine Axis in Breast Cancer Progression. Cancers. 2020. [PMC7226115](#)

Shi, Y; Riese, DJ; Shen, J. The Role of the CXCL12/CXCR4/CXCR7 Chemokine Axis in Cancer. Frontiers in pharmacology. 2020. [PMC7753359](#)

Stefani, C; Miricescu, D; Stanescu-Spinu, II; Nica, RI; Greabu, M; Totan, AR; Jinga, M. Growth Factors, PI3K/AKT/mTOR and MAPK Signaling Pathways in Colorectal Cancer Pathogenesis: Where Are We Now? International journal of molecular sciences. 2021.

[PMC8508474](#)

Wei, J; Wu, Y; Zhang, X; Sun, J; Li, J; Li, J; Yang, X; Qiao, H. Type 2 diabetes is more closely associated with risk of colorectal cancer based on elevated DNA methylation levels of ADCY5. Oncology letters. 2022. [PMC9178693](#)

Xu, J; Li, JQ; Chen, QL; Shestakova, EA; Misurin, VA; Pokrovsky, VS; Tchevkina, EM; Chen, HB; Song, H; Zhang, JY. Advances in Research on the Effects and Mechanisms of Chemokines and Their Receptors in Cancer. Frontiers in pharmacology. 2022.

[PMC9235028](#)

Xu, L; Wang, R; Ziegelbauer, J; Wu, WW; Shen, RF; Juhl, H; Zhang, Y; Pelosof, L; Rosenberg, AS. Transcriptome analysis of human colorectal cancer biopsies reveals extensive expression correlations among genes related to cell proliferation, lipid metabolism, immune response and collagen catabolism. Oncotarget. 2017. [PMC5650373](#)

Yao, J; Liang, LH; Zhang, Y; Ding, J; Tian, Q; Li, JJ; He, XH. GNAI1 Suppresses Tumor Cell Migration and Invasion and is Post-Transcriptionally Regulated by Mir-320a/c/d in Hepatocellular Carcinoma. Cancer biology & medicine. 2012. [PMC3643671](#)

Zaravinos, A; Roufas, C; Nagara, M; de Lucas Moreno, B; Oblovatskaya, M; Efsthathiades, C; Dimopoulos, C; Ayiomamitis, GD. Cytolytic activity correlates with the mutational burden and deregulated expression of immune checkpoints in colorectal cancer. *Journal of experimental & clinical cancer research : CR.* 2019. [PMC6701076](#)

Zhang, XF; Zhang, XL; Wang, YJ; Fang, Y; Li, ML; Liu, XY; Luo, HY; Tian, Y. The regulatory network of the chemokine CCL5 in colorectal cancer. *Annals of medicine.* 2023. [PMC10161960](#)

Zhao, R; Liu, J; Li, Z; Zhang, W; Wang, F; Zhang, B. Recent Advances in CXCL12/CXCR4 Antagonists and Nano-Based Drug Delivery Systems for Cancer Therapy. *Pharmaceutics.* 2022. [PMC9332179](#)

APPENDIX E FUNCTIONAL INTERFACE

LmRaC Command Line

The following is the function specification passed to GPT-4 in the LmRaC Command Line version allowing it to make return functional requests of LmRaC's RAG_{fun} module. In this case prompts to GPT-4 can result in one or more sequential replies requesting information about the expression of particular genes in the experimental results.

```

functions = [
    {
        'name': "exp_getGeneExpression",
        'description': "Return gene expression results for a list of
                        genes from an experiment",
        'parameters': {
            'type': "object",
            'properties': {
                'geneArray': {
                    'type': "array",
                    'items': {
                        'type': 'object',
                        'properties': {
                            'gene': {
                                'type': 'string',
                                'description':
                                    'Gene symbol'
                            }
                        },
                        'required': ['gene']
                    }
                },
                'experiment': {
                    'type': 'string',
                    'description': 'Name of experiment'
                }
            },
            'required': ['geneArray', 'experiment'],
        }
    }
]

```

LmRaC Web Application

For the LmRaC Web Application version this was simplified by implementing a pre-compiler for all functions. The same function is now specified by the user as follows:

```
FUNCTION getGeneExpression    "Return gene expression results for a list of genes
                                from an experiment"
PARAMETER geneArray:ARRAY    "List of genes"
ITEM gene:STRING    "Gene symbol"
PARAMETER filename:STRING*  "Gene expression results filename (optional)"
PARAMETER experiment:STRING "Name of experiment"
```

REFERENCES

1. Vaswani, A. *et al.* *Attention Is All You Need* arXiv: [1706.03762 \[cs\]](https://arxiv.org/abs/1706.03762). <http://arxiv.org/abs/1706.03762> (2024). Pre-published.
2. Zhao, W. X. *et al.* *A Survey of Large Language Models* arXiv: [2303.18223 \[cs\]](https://arxiv.org/abs/2303.18223). <http://arxiv.org/abs/2303.18223> (2024). Pre-published.
3. Minaee, S. *et al.* *Large Language Models: A Survey* arXiv: [2402.06196 \[cs\]](https://arxiv.org/abs/2402.06196). <http://arxiv.org/abs/2402.06196> (2024). Pre-published.
4. Fernández-Martín, J. C. *et al.* Gal3 Plays a Deleterious Role in a Mouse Model of Endotoxemia. *International Journal of Molecular Sciences* **23**, 1170. ISSN: 1422-0067. pmid: [35163089](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8835800/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8835800/> (2025) (Jan. 21, 2022).
5. Jeon, D., Hill, E. & McNeel, D. G. Toll-like Receptor Agonists as Cancer Vaccine Adjuvants. *Human Vaccines & Immunotherapeutics* **20**, 2297453. ISSN: 2164-5515. pmid: [38155525](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10760790/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10760790/> (2025) (2024).
6. Kuzmich, N. N. *et al.* TLR4 Signaling Pathway Modulators as Potential Therapeutics in Inflammation and Sepsis. *Vaccines* **5**, 34. ISSN: 2076-393X. pmid: [28976923](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5748601/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5748601/> (2025) (Oct. 4, 2017).
7. Park, Y. J. *et al.* The Effects of Etomidate on Expression of High Mobility Group Box 1 via the Nuclear Factor Kappa B Pathway in Rat Model of Sepsis. *The Libyan*

- Journal of Medicine* **18**, 2182683. ISSN: 1993-2820. pmid: [36855243](#). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9980160/> (2025) (2023).
8. Perrin-Cocon, L. *et al.* TLR4 Antagonist FP7 Inhibits LPS-induced Cytokine Production and Glycolytic Reprogramming in Dendritic Cells, and Protects Mice from Lethal Influenza Infection. *Scientific Reports* **7**, 40791. ISSN: 2045-2322. pmid: [28106157](#). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5247753/> (2025) (Jan. 20, 2017).
 9. Punch, E., Klein, J., Diaba-Nuhoho, P., Morawietz, H. & Garelnabi, M. Effects of PCSK9 Targeting: Alleviating Oxidation, Inflammation, and Atherosclerosis. *Journal of the American Heart Association: Cardiovascular and Cerebrovascular Disease* **11**, e023328. ISSN: 2047-9980. pmid: [35048716](#). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9238481/> (2025) (Jan. 20, 2022).
 10. Qiu, F., Zeng, C., Liu, Y., Pan, H. & Ke, C. J147 Ameliorates Sepsis-Induced Depressive-like Behaviors in Mice by Attenuating Neuroinflammation through Regulating the TLR4/NF-KB Signaling Pathway. *Journal of Molecular Histology* **54**, 725–738. ISSN: 1567-2379. pmid: [37676534](#). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10635911/> (2025) (2023).
 11. Shen, X., He, L. & Cai, W. Role of Lipopolysaccharides in the Inflammation and Pyroptosis of Alveolar Epithelial Cells in Acute Lung Injury and Acute Respiratory Distress Syndrome. *Journal of Inflammation Research* **17**, 5855–5869. ISSN: 1178-7031. pmid: [39228678](#). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11370780/> (2025) (Aug. 30, 2024).

12. Truhn, D., Reis-Filho, J. S. & Kather, J. N. Large Language Models Should Be Used as Scientific Reasoning Engines, Not Knowledge Databases. *Nature Medicine*, 1–2. ISSN: 1546-170X. <https://www.nature.com/articles/s41591-023-02594-z> (2023) (Oct. 18, 2023).
13. Craig, D. B. & Drăghici, S. *What's the Data Say? An LLM-based System for Interrogating Experimental Data* in 2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) (Dec. 2024), 1457–1462. <https://ieeexplore.ieee.org/document/10821725> (2025).
14. Craig, D. B. & Drăghici, S. LmRaC: A Functionally Extensible Tool for LLM Interrogation of User Experimental Results. *Bioinformatics* **40**, btae679. ISSN: 1367-4803. pmid: 39546375. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11645126/> (2025) (Nov. 15, 2024).
15. Craig, D. B. *Advancing Scientific Workflows: A Human-LLM Note-Taking System with Case-Based Reasoning* submitted to 2025 IEEE Conference on Artificial Intelligence (CAI). Jan. 15, 2025.
16. Craig, D. B. *A Human-LLM Note-Taking System with Case-Based Reasoning as Framework for Scientific Discovery* submitted to 2025 NAACL Workshop on AI and Scientific Discovery (AISD). Jan. 30, 2025.
17. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning Representations by Back-Propagating Errors. *Nature* **323**, 533–536. ISSN: 1476-4687. <https://www.nature.com/articles/323533a0> (2024) (6088 Oct. 1986).

18. Hochreiter, S. & Schmidhuber, J. Long Short-Term Memory. *Neural Computation* **9**, 1735–1780. ISSN: 0899-7667. <https://doi.org/10.1162/neco.1997.9.8.1735> (2024) (Nov. 15, 1997).
19. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* arXiv: [1810.04805 \[cs\]](https://arxiv.org/abs/1810.04805). <http://arxiv.org/abs/1810.04805> (2024). Pre-published.
20. Brown, T. B. *et al. Language Models Are Few-Shot Learners* arXiv: [2005.14165 \[cs\]](https://arxiv.org/abs/2005.14165). <http://arxiv.org/abs/2005.14165> (2024). Pre-published.
21. OpenAI. *GPT-4 Technical Report* arXiv: [2303.08774 \[cs\]](https://arxiv.org/abs/2303.08774). <http://arxiv.org/abs/2303.08774> (2023). Pre-published.
22. Azov, G., Pelc, T., Alon, A. F. & Kamhi, G. *Self-Improving Customer Review Response Generation Based on LLMs* arXiv: [2405.03845 \[cs\]](https://arxiv.org/abs/2405.03845). <http://arxiv.org/abs/2405.03845> (2024). Pre-published.
23. Chen, J. *et al. From Persona to Personalization: A Survey on Role-Playing Language Agents* arXiv: [2404.18231 \[cs\]](https://arxiv.org/abs/2404.18231). <http://arxiv.org/abs/2404.18231> (2024). Pre-published.
24. Pressman, S. M. *et al. Clinical and Surgical Applications of Large Language Models: A Systematic Review*. *Journal of Clinical Medicine* **13**, 3041. ISSN: 2077-0383. pmid: [38892752](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11172607/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11172607/> (2024) (May 22, 2024).

25. Nazi, Z. A. & Peng, W. *Large Language Models in Healthcare and Medical Domain: A Review* arXiv: [2401.06775 \[cs\]](https://arxiv.org/abs/2401.06775). <http://arxiv.org/abs/2401.06775> (2024). Pre-published.
26. Xu, H., Gan, W., Qi, Z., Wu, J. & Yu, P. S. *Large Language Models for Education: A Survey* arXiv: [2405.13001 \[cs\]](https://arxiv.org/abs/2405.13001). <http://arxiv.org/abs/2405.13001> (2024). Pre-published.
27. Lucas, H. C., Upperman, J. S. & Robinson, J. R. A Systematic Review of Large Language Models and Their Implications in Medical Education. *Medical Education*. ISSN: 1365-2923. pmid: [38639098](https://pubmed.ncbi.nlm.nih.gov/38639098/) (Apr. 19, 2024).
28. Roush, A. *et al. LLM as an Art Director (LaDi): Using LLMs to Improve Text-to-Media Generators* arXiv: [2311.03716 \[cs\]](https://arxiv.org/abs/2311.03716). <http://arxiv.org/abs/2311.03716> (2024). Pre-published.
29. Makridis, G., Oikonomou, A. & Koukos, V. *FairyLandAI: Personalized Fairy Tales Utilizing ChatGPT and DALLE-3* arXiv: [2407.09467 \[cs\]](https://arxiv.org/abs/2407.09467). <http://arxiv.org/abs/2407.09467> (2024). Pre-published.
30. Branch, B., Mirowski, P., Mathewson, K., Ppali, S. & Covaci, A. *Designing and Evaluating Dialogue LLMs for Co-Creative Improvised Theatre* arXiv: [2405.07111 \[cs\]](https://arxiv.org/abs/2405.07111). <http://arxiv.org/abs/2405.07111> (2024). Pre-published.
31. Martin, L., Whitehouse, N., Yiu, S., Catterson, L. & Perera, R. *Better Call GPT, Comparing Large Language Models Against Lawyers* version 1. arXiv: [2401.16212 \[cs\]](https://arxiv.org/abs/2401.16212). <http://arxiv.org/abs/2401.16212> (2024). Pre-published.

32. Sahu, G. *et al.* *InsightBench: Evaluating Business Analytics Agents Through Multi-Step Insight Generation* arXiv: [2407.06423 \[cs\]](https://arxiv.org/abs/2407.06423). <http://arxiv.org/abs/2407.06423> (2024). Pre-published.
33. Koncel-Kedziorski, R. *et al.* *BizBench: A Quantitative Reasoning Benchmark for Business and Finance* arXiv: [2311.06602 \[cs\]](https://arxiv.org/abs/2311.06602). <http://arxiv.org/abs/2311.06602> (2024). Pre-published.
34. Yu, J. *Retrieval Augmented Generation Integrated Large Language Models in Smart Contract Vulnerability Detection* arXiv: [2407.14838 \[cs\]](https://arxiv.org/abs/2407.14838). <http://arxiv.org/abs/2407.14838> (2024). Pre-published.
35. Puntoni, S., Ensing, M. & Bowers, J. How Marketers Can Adapt to LLM-Powered Search. *Harvard Business Review*. ISSN: 0017-8012. <https://hbr.org/2024/05/how-marketers-can-adapt-to-llm-powered-search> (2024) (May 24, 2024).
36. Yang, Q. *et al.* *SOMONITOR: Explainable Marketing Data Processing and Analysis with Large Language Models* arXiv: [2407.13117 \[cs\]](https://arxiv.org/abs/2407.13117). <http://arxiv.org/abs/2407.13117> (2024). Pre-published.
37. Dai, S.-C., Xiong, A. & Ku, L.-W. *LLM-in-the-loop: Leveraging Large Language Model for Thematic Analysis* arXiv: [2310.15100 \[cs\]](https://arxiv.org/abs/2310.15100). <http://arxiv.org/abs/2310.15100> (2024). Pre-published.
38. Gómez-Rodríguez, C. & Williams, P. *A Confederacy of Models: A Comprehensive Evaluation of LLMs on Creative Writing* arXiv: [2310.08433 \[cs\]](https://arxiv.org/abs/2310.08433). <http://arxiv.org/abs/2310.08433> (2024). Pre-published.

39. Davenport, T. H. & Mittal, N. How Generative AI Is Changing Creative Work. *Harvard Business Review*. ISSN: 0017-8012. <https://hbr.org/2022/11/how-generative-ai-is-changing-creative-work> (2024) (Nov. 14, 2022).
40. Romera-Paredes, B. *et al.* Mathematical Discoveries from Program Search with Large Language Models. *Nature*, 1–3. ISSN: 1476-4687. <https://www.nature.com/articles/s41586-023-06924-6> (2023) (Dec. 14, 2023).
41. Sweetser, P. *Large Language Models and Video Games: A Preliminary Scoping Review* arXiv: [2403.02613 \[cs\]](https://arxiv.org/abs/2403.02613). <http://arxiv.org/abs/2403.02613> (2024). Pre-published.
42. Agarwal, S., Laradji, I. H., Charlin, L. & Pal, C. *LitLLM: A Toolkit for Scientific Literature Review* version 1. arXiv: [2402.01788 \[cs\]](https://arxiv.org/abs/2402.01788). <http://arxiv.org/abs/2402.01788> (2024). Pre-published.
43. Dennstädt, F., Zink, J., Putora, P. M., Hastings, J. & Cihoric, N. Title and Abstract Screening for Literature Reviews Using Large Language Models: An Exploratory Study in the Biomedical Domain. *Systematic Reviews* **13**, 158. ISSN: 2046-4053. <https://doi.org/10.1186/s13643-024-02575-4> (2024) (June 15, 2024).
44. Gougherty, A. V. & Clipp, H. L. Testing the Reliability of an AI-based Large Language Model to Extract Ecological Information from the Scientific Literature. *npj Biodiversity* **3**, 1–5. ISSN: 2731-4243. <https://www.nature.com/articles/s44185-024-00043-9> (2024) (May 16, 2024).
45. Liang, W. *et al.* *Can Large Language Models Provide Useful Feedback on Research Papers? A Large-Scale Empirical Analysis* arXiv: [2310.01783 \[cs\]](https://arxiv.org/abs/2310.01783). <http://arxiv.org/abs/2310.01783> (2024). Pre-published.

46. Wang, J. *et al.* *Evaluating Large Language Models on Academic Literature Understanding and Review: An Empirical Study among Early-stage Scholars* in *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Association for Computing Machinery, New York, NY, USA, May 11, 2024), 1–18. ISBN: 9798400703300. <https://doi.org/10.1145/3613904.3641917> (2024).
47. Luo, X. *et al.* Potential Roles of Large Language Models in the Production of Systematic Reviews and Meta-Analyses. *Journal of Medical Internet Research* **26**, e56780. <https://www.jmir.org/2024/1/e56780> (2024) (June 25, 2024).
48. Lewis, P. *et al.* *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks* in *Advances in Neural Information Processing Systems* **33** (Curran Associates, Inc., 2020), 9459–9474. <https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html> (2023).
49. Guu, K., Lee, K., Tung, Z., Pasupat, P. & Chang, M.-W. *REALM: Retrieval-Augmented Language Model Pre-Training* arXiv: 2002.08909 [cs]. <http://arxiv.org/abs/2002.08909> (2024). Pre-published.
50. Gao, Y. *et al.* *Retrieval-Augmented Generation for Large Language Models: A Survey* arXiv: 2312.10997 [cs]. <http://arxiv.org/abs/2312.10997> (2023). Pre-published.
51. Wang, Z. *et al.* *Speculative RAG: Enhancing Retrieval Augmented Generation through Drafting* arXiv: 2407.08223 [cs]. <http://arxiv.org/abs/2407.08223> (2024). Pre-published.

52. Zhao, Y., Cao, H., Zhao, X. & Ou, Z. *An Empirical Study of Retrieval-Augmented Generation with Chain-of-Thought* arXiv: [2407.15569 \[cs\]](https://arxiv.org/abs/2407.15569). <http://arxiv.org/abs/2407.15569> (2024). Pre-published.
53. Zamani, H. & Bendersky, M. *Stochastic RAG: End-to-End Retrieval-Augmented Generation through Expected Utility Maximization* arXiv: [2405 . 02816 \[cs\]](https://arxiv.org/abs/2405.02816). <http://arxiv.org/abs/2405.02816> (2024). Pre-published.
54. Zhong, Z., Liu, H., Cui, X., Zhang, X. & Qin, Z. *Mix-of-Granularity: Optimize the Chunking Granularity for Retrieval-Augmented Generation* arXiv: [2406.00456 \[cs\]](https://arxiv.org/abs/2406.00456). <http://arxiv.org/abs/2406.00456> (2024). Pre-published.
55. Setty, S., Thakkar, H., Lee, A., Chung, E. & Vidra, N. *Improving Retrieval for RAG Based Question Answering Models on Financial Documents* arXiv: [2404.07221 \[cs, q-fin\]](https://arxiv.org/abs/2404.07221). <http://arxiv.org/abs/2404.07221> (2024). Pre-published.
56. Juvekar, K. & Purwar, A. *Introducing a New Hyper-Parameter for RAG: Context Window Utilization* arXiv: [2407.19794 \[cs\]](https://arxiv.org/abs/2407.19794). <http://arxiv.org/abs/2407.19794> (2024). Pre-published.
57. Akesson, S. & Santos, F. A. *Clustered Retrieved Augmented Generation (CRAG)* arXiv: [2406.00029 \[cs\]](https://arxiv.org/abs/2406.00029). <http://arxiv.org/abs/2406.00029> (2024). Pre-published.
58. Chen, R. *et al. Multi-Level Querying Using A Knowledge Pyramid* arXiv: [2407.21276 \[cs\]](https://arxiv.org/abs/2407.21276). <http://arxiv.org/abs/2407.21276> (2024). Pre-published.
59. IntelLabs/RAGFoundry Intel Labs, Aug. 7, 2024. <https://github.com/IntelLabs/RAGFoundry> (2024).

60. Wooldridge, M. *An Introduction to MultiAgent Systems* 2nd edition. 488 pp. ISBN: 978-0-470-51946-2 (Wiley, Chichester, June 22, 2009).
61. Norvig, P. & Russell, S. *Artificial Intelligence: A Modern Approach, Global Edition* 4th edition. 1166 pp. ISBN: 978-1-292-40113-3 (Pearson, Harlow, May 13, 2021).
62. Significant Gravitas. *AutoGPT* Aug. 3, 2024. <https://github.com/Significant-Gravitas/AutoGPT> (2024).
63. Wu, Q. *et al.* *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation* arXiv: 2308.08155 [cs]. <http://arxiv.org/abs/2308.08155> (2024). Pre-published.
64. Qian, C. *et al.* *ChatDev: Communicative Agents for Software Development* arXiv: 2307.07924 [cs]. <http://arxiv.org/abs/2307.07924> (2024). Pre-published.
65. *Embodied LLM Agents Learn to Cooperate in Organized Teams* <https://arxiv.org/html/2403.12482v1> (2024).
66. Abdelnabi, S., Gomaa, A., Sivaprasad, S., Schönherr, L. & Fritz, M. *Cooperation, Competition, and Maliciousness: LLM-Stakeholders Interactive Negotiation* arXiv: 2309.17234 [cs]. <http://arxiv.org/abs/2309.17234> (2024). Pre-published.
67. Yao, S. *et al.* *ReAct: Synergizing Reasoning and Acting in Language Models* arXiv: 2210.03629 [cs]. <http://arxiv.org/abs/2210.03629> (2024). Pre-published.
68. Aksitov, R. *et al.* *ReST Meets ReAct: Self-Improvement for Multi-Step Reasoning LLM Agent* arXiv: 2312.10003 [cs]. <http://arxiv.org/abs/2312.10003> (2023). Pre-published.

69. Liu, W. *et al.* *K-BERT: Enabling Language Representation with Knowledge Graph* arXiv: 1909.07606 [cs]. <http://arxiv.org/abs/1909.07606> (2024). Pre-published.
70. Zhang, Z. *et al.* *ERNIE: Enhanced Language Representation with Informative Entities* arXiv: 1905.07129 [cs]. <http://arxiv.org/abs/1905.07129> (2024). Pre-published.
71. Zhu, Y. *et al.* *LLMs for Knowledge Graph Construction and Reasoning: Recent Capabilities and Future Opportunities* arXiv: 2305.13168 [cs]. <http://arxiv.org/abs/2305.13168> (2024). Pre-published.
72. Ma, S. *et al.* *Think-on-Graph 2.0: Deep and Interpretable Large Language Model Reasoning with Knowledge Graph-guided Retrieval* arXiv: 2407.10805 [cs]. <http://arxiv.org/abs/2407.10805> (2024). Pre-published.
73. Lee, J., Wang, Y., Li, J. & Zhang, M. *Multimodal Reasoning with Multimodal Knowledge Graph* arXiv: 2406.02030 [cs]. <http://arxiv.org/abs/2406.02030> (2024). Pre-published.
74. Joshi, A. *et al.* *REAPER: Reasoning Based Retrieval Planning for Complex RAG Systems* arXiv: 2407.18553 [cs]. <http://arxiv.org/abs/2407.18553> (2024). Pre-published.
75. Wei, J. *et al.* *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models* arXiv: 2201.11903 [cs]. <http://arxiv.org/abs/2201.11903> (2024). Pre-published.
76. Yao, S. *et al.* *Tree of Thoughts: Deliberate Problem Solving with Large Language Models* arXiv: 2305.10601 [cs]. <http://arxiv.org/abs/2305.10601> (2024). Pre-published.

77. Shen, Y. *et al.* *HuggingGPT: Solving AI Tasks with ChatGPT and Its Friends in Hugging Face* arXiv: [2303.17580 \[cs\]](https://arxiv.org/abs/2303.17580). <http://arxiv.org/abs/2303.17580> (2024). Pre-published.
78. *Hugging Face – The AI Community Building the Future*. <https://huggingface.co/> (2024).
79. Lu, P. *et al.* *Chameleon: Plug-and-Play Compositional Reasoning with Large Language Models* arXiv: [2304.09842 \[cs\]](https://arxiv.org/abs/2304.09842). <http://arxiv.org/abs/2304.09842> (2024). Pre-published.
80. Lu, P. *et al.* *Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering* arXiv: [2209.09513 \[cs\]](https://arxiv.org/abs/2209.09513). <http://arxiv.org/abs/2209.09513> (2024). Pre-published.
81. Lu, P. *et al.* *Dynamic Prompt Learning via Policy Gradient for Semi-structured Mathematical Reasoning* arXiv: [2209.14610 \[cs\]](https://arxiv.org/abs/2209.14610). <http://arxiv.org/abs/2209.14610> (2024). Pre-published.
82. Plaat, A. *et al.* *Reasoning with Large Language Models, a Survey* arXiv: [2407.11511 \[cs\]](https://arxiv.org/abs/2407.11511). <http://arxiv.org/abs/2407.11511> (2024). Pre-published.
83. Madaan, A. *et al.* *Self-Refine: Iterative Refinement with Self-Feedback* arXiv: [2303.17651 \[cs\]](https://arxiv.org/abs/2303.17651). <http://arxiv.org/abs/2303.17651> (2024). Pre-published.
84. Shinn, N. *et al.* *Reflexion: Language Agents with Verbal Reinforcement Learning* arXiv: [2303.11366 \[cs\]](https://arxiv.org/abs/2303.11366). <http://arxiv.org/abs/2303.11366> (2024). Pre-published.

85. Liang, X. *et al.* *Internal Consistency and Self-Feedback in Large Language Models: A Survey* arXiv: [2407.14507 \[cs\]](https://arxiv.org/abs/2407.14507). <http://arxiv.org/abs/2407.14507> (2024). Pre-published.
86. Patil, S. G., Zhang, T., Wang, X. & Gonzalez, J. E. *Gorilla: Large Language Model Connected with Massive APIs* arXiv: [2305.15334 \[cs\]](https://arxiv.org/abs/2305.15334). <http://arxiv.org/abs/2305.15334> (2024). Pre-published.
87. Yang, Z. *et al.* *MM-REACT: Prompting ChatGPT for Multimodal Reasoning and Action* arXiv: [2303.11381 \[cs\]](https://arxiv.org/abs/2303.11381). <http://arxiv.org/abs/2303.11381> (2024). Pre-published.
88. Chen, S. *et al.* *Advancing Tool-Augmented Large Language Models: Integrating Insights from Errors in Inference Trees* arXiv: [2406.07115 \[cs\]](https://arxiv.org/abs/2406.07115). <http://arxiv.org/abs/2406.07115> (2024). Pre-published.
89. Guo, Z. *et al.* *StableToolBench: Towards Stable Large-Scale Benchmarking on Tool Learning of Large Language Models* arXiv: [2403.07714 \[cs\]](https://arxiv.org/abs/2403.07714). <http://arxiv.org/abs/2403.07714> (2024). Pre-published.
90. Girhepuje, S. *et al.* *RE-GAINS & EnChAnT: Intelligent Tool Manipulation Systems For Enhanced Query Responses* arXiv: [2401.15724 \[cs\]](https://arxiv.org/abs/2401.15724). <http://arxiv.org/abs/2401.15724> (2024). Pre-published.
91. Kong, Y. *et al.* *TPTU-v2: Boosting Task Planning and Tool Usage of Large Language Model-based Agents in Real-world Systems* arXiv: [2311.11315 \[cs\]](https://arxiv.org/abs/2311.11315). <http://arxiv.org/abs/2311.11315> (2024). Pre-published.

92. Zhuang, Y. *et al.* *ToolChain*: Efficient Action Space Navigation in Large Language Models with A* Search* arXiv: [2310.13227 \[cs\]](https://arxiv.org/abs/2310.13227). <http://arxiv.org/abs/2310.13227> (2024). Pre-published.
93. Newell, A. & Simon, H. A. *Human Problem Solving* Reprint ed. edition. 938 pp. ISBN: 978-1-64837-194-3 (Echo Point Books & Media, LLC, Feb. 14, 2023).
94. A, L. P. H. N. D. *Human Information Processing: Introduction to Psychology by Lindsay Peter H. Norman Donald A. (1972-07-01) Hardcover* () .
95. Kahneman, D. *Thinking, Fast and Slow* 1st edition. 512 pp. ISBN: 978-0-374-27563-1 (Farrar, Straus and Giroux, New York, Oct. 25, 2011).
96. Klahr, D. & Simon, H. A. What Have Psychologists (And Others) Discovered About the Process of Scientific Discovery? *Current Directions in Psychological Science* **10**, 75–79. ISSN: 0963-7214, 1467-8721. <http://journals.sagepub.com/doi/10.1111/1467-8721.00119> (2024) (June 2001).
97. Klahr, D. & Simon, H. A. Studies of Scientific Discovery: Complementary Approaches and Convergent Findings. *Psychological Bulletin* **125**, 524 (1999).
98. Simon, H. A. & Laird, J. E. *The Sciences of the Artificial, Reissue of the Third Edition with a New Introduction by John Laird* Reissue edition. 256 pp. ISBN: 978-0-262-53753-7 (The MIT Press, Cambridge, Massachusetts, Aug. 13, 2019).
99. Hu, X., Storks, S., Lewis, R. L. & Chai, J. *In-Context Analogical Reasoning with Pre-Trained Language Models* arXiv: [2305.17626 \[cs\]](https://arxiv.org/abs/2305.17626). <http://arxiv.org/abs/2305.17626> (2023). Pre-published.

100. Musker, S., Duchnowski, A., Millière, R. & Pavlick, E. *Semantic Structure-Mapping in LLM and Human Analogical Reasoning* arXiv: [2406.13803 \[cs\]](https://arxiv.org/abs/2406.13803). <http://arxiv.org/abs/2406.13803> (2024). Pre-published.
101. Qin, C. *et al.* *Relevant or Random: Can LLMs Truly Perform Analogical Reasoning?* arXiv: [2404.12728 \[cs\]](https://arxiv.org/abs/2404.12728). <http://arxiv.org/abs/2404.12728> (2024). Pre-published.
102. Tong, X., Choenni, R., Lewis, M. & Shutova, E. *Metaphor Understanding Challenge Dataset for LLMs* arXiv: [2403.11810 \[cs\]](https://arxiv.org/abs/2403.11810). <http://arxiv.org/abs/2403.11810> (2024). Pre-published.
103. Yu, J., He, R. & Ying, R. *Thought Propagation: An Analogical Approach to Complex Reasoning with Large Language Models* arXiv: [2310.03965 \[cs\]](https://arxiv.org/abs/2310.03965). <http://arxiv.org/abs/2310.03965> (2024). Pre-published.
104. Luo, R. *et al.* BioGPT: Generative Pre-trained Transformer for Biomedical Text Generation and Mining. *Briefings in Bioinformatics* **23**, bbac409. ISSN: 1467-5463, 1477-4054. arXiv: [2210.10341 \[cs\]](https://arxiv.org/abs/2210.10341). <http://arxiv.org/abs/2210.10341> (2024) (Nov. 19, 2022).
105. Lee, J. *et al.* BioBERT: A Pre-Trained Biomedical Language Representation Model for Biomedical Text Mining. *Bioinformatics* **36**, 1234–1240. ISSN: 1367-4803. <https://doi.org/10.1093/bioinformatics/btz682> (2024) (Feb. 15, 2020).
106. Beltagy, I., Lo, K. & Cohan, A. *SciBERT: A Pretrained Language Model for Scientific Text* arXiv: [1903.10676 \[cs\]](https://arxiv.org/abs/1903.10676). <http://arxiv.org/abs/1903.10676> (2024). Pre-published.

107. Gu, Y. *et al.* Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Transactions on Computing for Healthcare* **3**, 2:1–2:23. <https://doi.org/10.1145/3458754> (2024) (Oct. 15, 2021).
108. Bhattacharjee, B. *et al.* INDUS: Effective and Efficient Language Models for Scientific Applications arXiv: [2405.10725 \[cs\]](https://arxiv.org/abs/2405.10725). <http://arxiv.org/abs/2405.10725> (2024). Pre-published.
109. Sanabria, M., Hirsch, J., Joubert, P. M. & Poetsch, A. R. DNA Language Model GROVER Learns Sequence Context in the Human Genome. *Nature Machine Intelligence*, 1–13. ISSN: 2522-5839. <https://www.nature.com/articles/s42256-024-00872-0> (2024) (July 23, 2024).
110. Benegas, G., Ye, C., Albors, C., Li, J. C. & Song, Y. S. *Genomic Language Models: Opportunities and Challenges* arXiv: [2407.11435 \[cs, q-bio, stat\]](https://arxiv.org/abs/2407.11435). <http://arxiv.org/abs/2407.11435> (2024). Pre-published.
111. Lála, J. *et al.* PaperQA: Retrieval-Augmented Generative Agent for Scientific Research version 2. arXiv: [2312.07559 \[cs\]](https://arxiv.org/abs/2312.07559). <http://arxiv.org/abs/2312.07559> (2024). Pre-published.
112. Soong, D. *et al.* Improving Accuracy of GPT-3/4 Results on Biomedical Data Using a Retrieval-Augmented Language Model arXiv: [2305.17116 \[cs\]](https://arxiv.org/abs/2305.17116). <http://arxiv.org/abs/2305.17116> (2024). Pre-published.
113. Zakka, C. *et al.* Almanac: Retrieval-Augmented Language Models for Clinical Medicine. *Research Square*, rs.3.rs–2883198. pmid: [37205549](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10187428/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10187428/> (2024) (May 2, 2023).

114. Pelletier, A. R. *et al.* *Explainable Biomedical Hypothesis Generation via Retrieval Augmented Generation Enabled Large Language Models* arXiv: [2407.12888 \[cs\]](https://arxiv.org/abs/2407.12888). <http://arxiv.org/abs/2407.12888> (2024). Pre-published.
115. O'Donoghue, O. *et al.* *BioPlanner: Automatic Evaluation of LLMs on Protocol Planning in Biology* arXiv: [2310.10632 \[cs\]](https://arxiv.org/abs/2310.10632). <http://arxiv.org/abs/2310.10632> (2024). Pre-published.
116. Song, S. L. *et al.* *DeepSpeed4Science Initiative: Enabling Large-Scale Scientific Discovery through Sophisticated AI System Technologies* arXiv: [2310.04610 \[cs\]](https://arxiv.org/abs/2310.04610). <http://arxiv.org/abs/2310.04610> (2024). Pre-published.
117. Laurent, J. M. *et al.* *LAB-Bench: Measuring Capabilities of Language Models for Biology Research* arXiv: [2407.10362 \[cs\]](https://arxiv.org/abs/2407.10362). <http://arxiv.org/abs/2407.10362> (2024). Pre-published.
118. Feng, K. *et al.* *SciKnowEval: Evaluating Multi-level Scientific Knowledge of Large Language Models* arXiv: [2406.09098 \[cs\]](https://arxiv.org/abs/2406.09098). <http://arxiv.org/abs/2406.09098> (2024). Pre-published.
119. Shi, T. *et al.* *Stochastic Parrots or ICU Experts? Large Language Models in Critical Care Medicine: A Scoping Review* arXiv: [2407.19256 \[cs\]](https://arxiv.org/abs/2407.19256). <http://arxiv.org/abs/2407.19256> (2024). Pre-published.
120. Zhou, H.-Y., Adithan, S., Acosta, J. N., Topol, E. J. & Rajpurkar, P. *A Generalist Learner for Multifaceted Medical Image Interpretation* arXiv: [2405.07988 \[cs\]](https://arxiv.org/abs/2405.07988). <http://arxiv.org/abs/2405.07988> (2024). Pre-published.

121. Ferber, D. *et al.* *End-To-End Clinical Trial Matching with Large Language Models* arXiv: 2407.13463 [cs]. <http://arxiv.org/abs/2407.13463> (2024). Pre-published.
122. *Leveraging Large Language Models for Decision Support in Personalized Oncology | Oncology | JAMA Network Open | JAMA Network* <https://jamanetwork.com/journals/jamanetworkopen/fullarticle/2812097> (2024).
123. Li, T. *et al.* CancerGPT for Few Shot Drug Pair Synergy Prediction Using Large Pretrained Language Models. *npj Digital Medicine* 7, 1–10. ISSN: 2398-6352. <https://www.nature.com/articles/s41746-024-01024-9> (2024) (Feb. 19, 2024).
124. Li, L. *et al.* *A Scoping Review of Using Large Language Models (LLMs) to Investigate Electronic Health Records (EHRs)* arXiv: 2405.03066 [cs]. <http://arxiv.org/abs/2405.03066> (2024). Pre-published.
125. Van Veen, D. *et al.* Adapted Large Language Models Can Outperform Medical Experts in Clinical Text Summarization. *Nature Medicine* 30, 1134–1142. ISSN: 1546-170X. <https://www.nature.com/articles/s41591-024-02855-5> (2024) (Apr. 2024).
126. Ke, Y. *et al.* *Development and Testing of Retrieval Augmented Generation in Large Language Models – A Case Study Report* arXiv: 2402.01733 [cs]. <http://arxiv.org/abs/2402.01733> (2024). Pre-published.
127. McDuff, D. *et al.* *Towards Accurate Differential Diagnosis with Large Language Models* arXiv: 2312.00164 [cs]. <http://arxiv.org/abs/2312.00164> (2024). Pre-published.

128. Mehandru, N. *et al.* Evaluating Large Language Models as Agents in the Clinic. *npj Digital Medicine* **7**, 1–3. ISSN: 2398-6352. <https://www.nature.com/articles/s41746-024-01083-y> (2024) (Apr. 3, 2024).
129. *Large Language Models Encode Clinical Knowledge* | *Nature* <https://www.nature.com/articles/s41586-023-06291-2> (2024).
130. Li, J. *et al.* Integrated Image-Based Deep Learning and Language Models for Primary Diabetes Care. *Nature Medicine*, 1–11. ISSN: 1546-170X. <https://www.nature.com/articles/s41591-024-03139-8> (2024) (July 19, 2024).
131. Kim, J. *et al.* *Patient Perspectives on Large Language Model Responses to Patient Messages* <https://papers.ssrn.com/abstract=4867523> (2024). Pre-published.
132. *Large Language Models in Medicine* | *Nature Medicine* <https://www.nature.com/articles/s41591-023-02448-8> (2024).
133. Perez-Lopez, R., Ghaffari Laleh, N., Mahmood, F. & Kather, J. N. A Guide to Artificial Intelligence for Cancer Researchers. *Nature Reviews Cancer* **24**, 427–441. ISSN: 1474-1768. <https://www.nature.com/articles/s41568-024-00694-7> (2024) (June 2024).
134. *The Breakthrough of Large Language Models Release for Medical Applications: 1-Year Timeline and Perspectives* | *Journal of Medical Systems* <https://link.springer.com/article/10.1007/s10916-024-02045-3> (2024).
135. Farquhar, S., Kossen, J., Kuhn, L. & Gal, Y. Detecting Hallucinations in Large Language Models Using Semantic Entropy. *Nature* **630**, 625–630. ISSN: 1476-4687. <https://www.nature.com/articles/s41586-024-07421-0> (2024) (June 2024).

136. Nowogrodzki, J. ChatGPT for Science: How to Talk to Your Data. *Nature* **631**, 924–925. <https://www.nature.com/articles/d41586-024-02386-6> (2024) (July 22, 2024).
137. Almarie, B., Teixeira, P. E. P., Pacheco-Barrios, K., Rossetti, C. A. & Fregni, F. Editorial – The Use of Large Language Models in Science: Opportunities and Challenges. *Principles and practice of clinical research* (2015) **9**, 1–4. ISSN: 2378-1890. pmid: 37693832. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10485814/> (2024) (July 10, 2023).
138. Sanderson, K. GPT-4 Is Here: What Scientists Think. *Nature* **615**, 773–773. <https://www.nature.com/articles/d41586-023-00816-5> (2023) (7954 Mar. 16, 2023).
139. Tang, X. *et al.* Prioritizing Safeguarding Over Autonomy: Risks of LLM Agents for Science arXiv: [2402.04247 \[cs\]](https://arxiv.org/abs/2402.04247). <http://arxiv.org/abs/2402.04247> (2024). Pre-published.
140. Haltaufderheide, J. & Ranisch, R. The Ethics of ChatGPT in Medicine and Healthcare: A Systematic Review on Large Language Models (LLMs). *npj Digital Medicine* **7**, 1–11. ISSN: 2398-6352. <https://www.nature.com/articles/s41746-024-01157-x> (2024) (July 8, 2024).
141. Birhane, A., Kasirzadeh, A., Leslie, D. & Wachter, S. Science in the Age of Large Language Models. *Nature Reviews Physics* **5**, 277–280. ISSN: 2522-5820. <https://www.nature.com/articles/s42254-023-00581-4> (2024) (May 2023).

142. Bran, A. M. *et al.* *ChemCrow: Augmenting Large-Language Models with Chemistry Tools* arXiv: 2304.05376 [physics, stat]. <http://arxiv.org/abs/2304.05376> (2024). Pre-published.
143. Yoshikawa, N. *et al.* Large Language Models for Chemistry Robotics. *Autonomous Robots* **47**, 1057–1086. ISSN: 1573-7527. <https://doi.org/10.1007/s10514-023-10136-2> (2024) (Dec. 1, 2023).
144. Boiko, D. A., MacKnight, R., Kline, B. & Gomes, G. Autonomous Chemical Research with Large Language Models. *Nature* **624**, 570–578. ISSN: 1476-4687. <https://www.nature.com/articles/s41586-023-06792-0> (2024) (Dec. 2023).
145. He, J. *et al.* *Control Risk for Potential Misuse of Artificial Intelligence in Science* arXiv: 2312.06632 [cs]. <http://arxiv.org/abs/2312.06632> (2024). Pre-published.
146. Gao, H. & Zhang, Y. *Memory Sharing for Large Language Model Based Agents* arXiv: 2404.09982 [cs]. <http://arxiv.org/abs/2404.09982> (2024). Pre-published.
147. [2407.09450v1] *Human-like Episodic Memory for Infinite Context LLMs* <https://arxiv.org/abs/2407.09450v1> (2024).
148. [1605.06676] *Learning to Communicate with Deep Multi-Agent Reinforcement Learning* <https://arxiv.org/abs/1605.06676> (2024).
149. [1410.5401] *Neural Turing Machines* <https://arxiv.org/abs/1410.5401> (2024).
150. *MemoryBank: Enhancing Large Language Models with Long-Term Memory* | *Proceedings of the AAAI Conference on Artificial Intelligence* <https://ojs.aaai.org/index.php/AAAI/article/view/29946> (2024).

151. Collaborative Intelligence: Humans and AI Are Joining Forces. *Harvard Business Review*. ISSN: 0017-8012. <https://hbr.org/2018/07/collaborative-intelligence-humans-and-ai-are-joining-forces> (2024) (July 1, 2018).
152. Mosqueira-Rey, E., Hernández-Pereira, E., Alonso-Ríos, D., Bobes-Bascarán, J. & Fernández-Leal, Á. Human-in-the-Loop Machine Learning: A State of the Art. *Artificial Intelligence Review* **56**, 3005–3054. ISSN: 1573-7462. <https://doi.org/10.1007/s10462-022-10246-w> (2024) (Apr. 1, 2023).
153. Panait, L. & Luke, S. Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems* **11**, 387–434. ISSN: 1573-7454. <https://doi.org/10.1007/s10458-005-2631-2> (2024) (Nov. 1, 2005).
154. PubMed <https://pubmed.ncbi.nlm.nih.gov/> (2024).
155. iPathwayGuide <https://ipathwayguide.advaitabio.com/dashboard> (2024).
156. Project Jupyter <https://jupyter.org> (2024).
157. Google Colab <https://colab.research.google.com/?authuser=1> (2024).
158. Obsidian - Sharpen Your Thinking <https://obsidian.md/> (2024).
159. Visual Studio Code - Code Editing. Redefined <https://code.visualstudio.com/> (2025).
160. Welcome to Python.Org Python.org. <https://www.python.org/> (2024).
161. OpenAI Platform <https://platform.openai.com> (2024).
162. Pinecone: The Vector Database to Build Knowledgeable AI <https://www.pinecone.io/> (2024).
163. Firestore Firebase. <https://firebase.google.com/docs/firestore> (2024).

164. *Welcome to Flask — Flask Documentation (3.1.x)* <https://flask.palletsprojects.com/en/stable/> (2025).
165. *Docker: Accelerated Container Application Development* <https://www.docker.com/> (2025).
166. *Neo4j Graph Database & Analytics – The Leader in Graph Databases* Graph Database & Analytics. <https://neo4j.com/> (2025).
167. *Openai/Whisper* OpenAI, Jan. 19, 2025. <https://github.com/openai/whisper> (2025).
168. *Text to Speech - OpenAI API* <https://platform.openai.com/docs/guides/text-to-speech> (2025).
169. *Pygame/Pygame* pygame, Jan. 19, 2025. <https://github.com/pygame/pygame> (2025).
170. *PMC OA Web Service API* PubMed Central (PMC). <https://www.ncbi.nlm.nih.gov/pmc/tools/oa-service/> (2024).
171. *Beautiful Soup Documentation — Beautiful Soup 4.4.0 Documentation* <https://beautiful-soup-4.readthedocs.io/en/latest/#> (2024).
172. *R: The R Project for Statistical Computing* <https://www.r-project.org/> (2024).
173. *Ggplot2* <https://ggplot2.tidyverse.org/> (2024).
174. *DALL·E 3* <https://openai.com/index/dall-e-3/> (2025).
175. *The Perl Programming Language - Www.Perl.Org* <https://www.perl.org/> (2025).

176. Will ChatGPT Transform Healthcare? *Nature Medicine* **29**, 505–506. ISSN: 1546-170X. <https://www.nature.com/articles/s41591-023-02289-5> (2023) (3 Mar. 2023).
177. Sanderson, K. GPT-4 Is Here: What Scientists Think. *Nature* **615**, 773–773. <https://www.nature.com/articles/d41586-023-00816-5> (2023) (7954 Mar. 16, 2023).
178. Zhang, Y. *et al.* *Siren’s Song in the AI Ocean: A Survey on Hallucination in Large Language Models* arXiv: [2309.01219 \[cs\]](https://arxiv.org/abs/2309.01219). <http://arxiv.org/abs/2309.01219> (2024). Pre-published.
179. Huang, L. *et al.* *A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions* arXiv: [2311.05232 \[cs\]](https://arxiv.org/abs/2311.05232). <http://arxiv.org/abs/2311.05232> (2024). Pre-published.
180. Wysocka, M., Wysocki, O., Delmas, M., Mutel, V. & Freitas, A. *Large Language Models, Scientific Knowledge and Factuality: A Systematic Analysis in Antibiotic Discovery* arXiv: [2305.17819 \[cs\]](https://arxiv.org/abs/2305.17819). <http://arxiv.org/abs/2305.17819> (2024). Pre-published.
181. Bhattacharyya, M., Miller, V. M., Bhattacharyya, D. & Miller, L. E. High Rates of Fabricated and Inaccurate References in ChatGPT-Generated Medical Content. *Cureus* **15**, e39238. ISSN: 2168-8184. pmid: 37337480. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10277170/> (2024).
182. Google Scholar <https://scholar.google.com/> (2024).

183. Goldman, M. J. *et al.* Visualizing and Interpreting Cancer Genomics Data via the Xena Platform. *Nature biotechnology* **38**, 675. pmid: [32444850](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7386072/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7386072/> (2024) (June 2020).
184. Muzny, D. M. *et al.* Comprehensive Molecular Characterization of Human Colon and Rectal Cancer. *Nature* **487**, 330–337. ISSN: 0028-0836. pmid: [22810696](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC22810696) (2012).
185. Draghici, S. *et al.* A Systems Biology Approach for Pathway Level Analysis. *Genome Research* **17**, 1537–1545. ISSN: 1088-9051. pmid: [17785539](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1987343/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1987343/> (2024) (Oct. 2007).
186. Donato, M. *et al.* Analysis and Correction of Crosstalk Effects in Pathway Analysis. *Genome Research* **23**, 1885–1893. ISSN: 1088-9051. pmid: [23934932](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3814888/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3814888/> (2024) (Nov. 2013).
187. KEGG PATHWAY Database <https://www.genome.jp/kegg/pathway.html> (2024).
188. MeSH - NCBI <https://www.ncbi.nlm.nih.gov/mesh/> (2024).
189. HUGO Gene Nomenclature Committee <https://www.genenames.org/download/statistics-and-files/> (2024).
190. Touvron, H. *et al.* Llama 2: Open Foundation and Fine-Tuned Chat Models arXiv: [2307.09288 \[cs\]](https://arxiv.org/abs/2307.09288). <http://arxiv.org/abs/2307.09288> (2024). Pre-published.
191. Anil, R. *et al.* PaLM 2 Technical Report arXiv: [2305.10403 \[cs\]](https://arxiv.org/abs/2305.10403). <http://arxiv.org/abs/2305.10403> (2024). Pre-published.
192. in. Wikipedia (July 5, 2024). <https://en.wikipedia.org/w/index.php?title=Zettelkasten&oldid=1232806331> (2024).

193. Kolodner, J. *Case-Based Reasoning* 668 pp. ISBN: 978-1-55860-237-3 (Morgan Kaufmann, San Mateo, CA, Sept. 15, 1993).
194. Watson, I. A *Case-Based Persistent Memory for a Large Language Model* arXiv: [2310.08842 \[cs\]](#). <http://arxiv.org/abs/2310.08842> (2024). Pre-published.
195. Schank, R. C. *Conceptual Information Processing* ISBN: 978-0-7204-2507-9 (Elsevier Science Ltd, Amsterdam, Jan. 1, 1975).
196. Schank, R. C. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People* 1st edition. 272 pp. ISBN: 978-0-521-24858-7 (Cambridge University Press, Cambridge, Jan. 28, 1983).
197. Kolodner, J. L. Reconstructive Memory: A Computer Model. *Cognitive Science* 7, 281–328. ISSN: 1551-6709. https://onlinelibrary.wiley.com/doi/10.1207/s15516709cog0704_2 (2025) (1983).
198. Wilkerson, K. & Leake, D. *On Implementing Case-Based Reasoning with Large Language Models* in *Case-Based Reasoning Research and Development* (eds Recio-Garcia, J. A., Orozco-del-Castillo, M. G. & Bridge, D.) (Springer Nature Switzerland, Cham, 2024), 404–417. ISBN: 978-3-031-63646-2.
199. Yang, R. *CaseGPT: A Case Reasoning Framework Based on Language Models and Retrieval-Augmented Generation* arXiv: [2407.07913 \[cs\]](#). <http://arxiv.org/abs/2407.07913> (2025). Pre-published.
200. Richter, M. M. & Weber, R. O. *Case-Based Reasoning: A Textbook* 2013th edition. 564 pp. ISBN: 978-3-642-40166-4 (Springer, Berlin, Heidelberg, Nov. 15, 2013).

201. Weitl-Harms, S., Hastings, J. & Powell, J. Toward Automated Knowledge Discovery in Case-Based Reasoning. *The International FLAIRS Conference Proceedings* 37. ISSN: 2334-0762. <https://journals.flvc.org/FLAIRS/article/view/135434> (2025) (May 13, 2024).
202. Edge, D. *et al.* *From Local to Global: A Graph RAG Approach to Query-Focused Summarization* arXiv: 2404.16130 [cs]. <http://arxiv.org/abs/2404.16130> (2025). Pre-published.
203. Peng, B. *et al.* *Graph Retrieval-Augmented Generation: A Survey* arXiv: 2408.08921 [cs]. <http://arxiv.org/abs/2408.08921> (2025). Pre-published.
204. Blei, D. M. Latent Dirichlet Allocation.
205. *Introducing ChatGPT Search* <https://openai.com/index/introducing-chatgpt-search/> (2025).
206. Watson, I. *Applying Case-Based Reasoning: Techniques for Enterprise Systems* 1st edition. 290 pp. ISBN: 978-1-55860-462-9 (Morgan Kaufmann, San Francisco, Calif, July 15, 1997).
207. Liu, X. *et al.* *Are LLMs Capable of Data-based Statistical and Causal Reasoning? Benchmarking Advanced Quantitative Reasoning with Data* arXiv: 2402.17644 [cs]. <http://arxiv.org/abs/2402.17644> (2025). Pre-published.
208. Majumder, B. P. *et al.* *DiscoveryBench: Towards Data-Driven Discovery with Large Language Models* arXiv: 2407.01725 [cs]. <http://arxiv.org/abs/2407.01725> (2025). Pre-published.

209. Chen, Z. *et al.* *ScienceAgentBench: Toward Rigorous Assessment of Language Agents for Data-Driven Scientific Discovery* arXiv: [2410.05080 \[cs\]](https://arxiv.org/abs/2410.05080). <http://arxiv.org/abs/2410.05080> (2025). Pre-published.
210. Rein, D. *et al.* *GPQA: A Graduate-Level Google-Proof Q&A Benchmark* arXiv: [2311.12022 \[cs\]](https://arxiv.org/abs/2311.12022). <http://arxiv.org/abs/2311.12022> (2025). Pre-published.
211. Chollet, F., Knoop, M., Kamradt, G. & Landers, B. *ARC Prize 2024: Technical Report* arXiv: [2412.04604 \[cs\]](https://arxiv.org/abs/2412.04604). <http://arxiv.org/abs/2412.04604> (2025). Pre-published.
212. Jones, N. AI Hallucinations Can't Be Stopped — but These Techniques Can Limit Their Damage. *Nature* **637**, 778–780. ISSN: 1476-4687. <https://www.nature.com/articles/d41586-025-00068-5> (2025) (Jan. 21, 2025).
213. Anishchenko, I. *et al.* De Novo Protein Design by Deep Network Hallucination. *Nature* **600**, 547–552. ISSN: 0028-0836. pmid: [34853475](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9293396/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9293396/> (2025) (Dec. 2021).
214. Watson, J. L. *et al.* De Novo Design of Protein Structure and Function with RFdiffusion. *Nature* **620**, 1089–1100. ISSN: 0028-0836. pmid: [37433327](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10468394/). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10468394/> (2025) (2023).
215. Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N. & Ganguli, S. *Deep Unsupervised Learning Using Nonequilibrium Thermodynamics* in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37* (JMLR.org, Lille, France, July 6, 2015), 2256–2265.
216. Ho, J., Jain, A. & Abbeel, P. *Denoising Diffusion Probabilistic Models* in *Advances in Neural Information Processing Systems* **33** (Curran Associates,

- Inc., 2020), 6840–6851. <https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html> (2025).
217. Ramesh, A. *et al.* Zero-Shot Text-to-Image Generation in *Proceedings of the 38th International Conference on Machine Learning* International Conference on Machine Learning (PMLR, July 1, 2021), 8821–8831. <https://proceedings.mlr.press/v139/ramesh21a.html> (2025).
218. Saharia, C. *et al.* Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *Advances in Neural Information Processing Systems* **35**, 36479–36494. https://proceedings.neurips.cc/paper_files/paper/2022/hash/ec795aeadae0b7d230fa35cbaf04c041-Abstract-Conference.html (2025) (Dec. 6, 2022).

INDEX

A

- agentic architecture 13, 183
 AI *see* artificial intelligence
 artificial intelligence
 human-centered 120, 140
 attention 7
 attention, self 7

B

- back-propagation 7
 BERT 7
 BioBERT 16
 BioGPT 16, 53
 biology 16, 26

C

- case-based reasoning 119, 122, 125, 130,
 141, 144, 179
 memory 130
 CBR *see* case-based reasoning
 chain-of-thought 14

D

- diffusion 186
 Docker 59, 81, 82

E

- embeddings *see* vector embeddings
 explainable AI 130

F

- feed-forward 7
 few-shot learning 9

G

- GPT-3 7, 53
 GPT-4 7, 53
 graph database 130, 180

H

- hallucination *see* large-language models
 human-in-the-loop 140

I

- iPathway Guide 44

J

- JSON 9

K

- knowledge graphs 13

L

- language, natural 7
 large-language models 7
 agents 13
 application domains 10
 as reasoning engines 34
 building 10
 collaboration 122, 141
 data structures 13
 domain specific 36
 drug discovery 16
 ethics 19
 feedback 15
 hallucination 18, 55, 148, 186
 health care 17
 history 9
 literature search 16, 150
 medicine 17
 planning 14
 prompts 9
 reflection 15, 159, 181
 response 9
 science 16
 tool use 15, 123, 141, 153
 transparency 19
 trust 59
 usage 9
 Llama 9, 53
 LLMs *see* large-language models
 LmRaC
 Command-Line 21, 33
 design 37
 domain knowledge 38, 45
 embeddings 39
 experiments 48
 feedback 37
 functions 42

model comparison	53
pathway analysis	44
performance	53
quality assessment	41
questions	46
RAG	50
sub-questions	39
usefulness metric	40, 50
vector embeddings	51
Obsidian note-taking ...	24, 118, 138
bootstrapping	122
case-based reasoning	122
collaboration	122
console	126
design	120
executive	125
innovations	130
problem solving, progressive ..	123
scientific workflow	119, 132
tools	135
useability	130
Web Application	23, 58
answers	94
architecture	61
commands	88
configuration	97
experiments	63, 67, 92, 115
functions	93, 115
homepage	87
indexes	65, 90, 115
installation	81
limitations	70
prerequisites	72
quick-start	73
REST API	69, 107, 158
troubleshooting	115
user-defined functions <i>see</i> functions	
workflow	64
long short-term memory	35
M	
memory	<i>see</i> case-based reasoning
model, language	<i>see</i> large-language models
N	
neural networks	7
O	
one-shot learning	9
P	
PaLM	9, 53
pathway analysis	26
Pinecone	<i>see</i> vector database
problem solving	
human	16
progressive	123
protein design	186
Q	
query-key-value	7
R	
RAG . <i>see</i> retrieval-augmented generation	
recurrent neural networks	7, 35
retrieval-augmented generation ..	10, 36, 56, 60, 130, 150
chunking	14
graphRAG	130
metadata	14, 39
stochastic	14
variations	12
S	
SciBERT	16
scientific discovery	16, 134, 138, 180
serverless	43
Softmax	7
T	
transformer architecture	7, 8, 35
U	
unsupervised learning	186
V	
vector database	43, 62
vector embeddings	51, 128
W	
word embeddings <i>see</i> vector embeddings	
X	
XAI	<i>see</i> explainable AI

YYAML [125](#)**Z**zero-shot learning [9](#)

ABSTRACT

LEVERAGING LARGE-LANGUAGE MODELS AS COLLABORATIVE REASONING PARTNERS TO ENHANCE SCIENTIFIC WORKFLOW

by

DOUGLAS B. CRAIG

May 2025

Advisor: Dr. Sorin Drăghici

Major: Computer Science (Bioinformatics & Computational Biology)

Degree: Doctor of Philosophy

The rise of Large Language Models (LLMs) has transformed artificial intelligence, offering advanced capabilities in text generation, natural language understanding, and multi-modal interactions. However, their use as standalone tools or as perceived repositories of static knowledge has limited their potential in real-world applications, especially in critical domains like healthcare and scientific research, where transparency, explainability, and accountability are paramount. This research addresses these limitations by conceptualizing LLMs as reasoning engines within a hybrid framework that integrates retrieval-augmented generation (RAG) and case-based reasoning (CBR) within a note-taking application.

The study introduces a novel system, LmRaC, designed to enhance the reliability, explainability, and collaborative utility of LLMs. LmRaC is embedded within the Obsidian note-taking application, offering a dynamic environment where users can interact with LLM-powered tools to generate traceable and goal-oriented solutions. By combining authoritative RAG-based information retrieval with a collaborative CBR framework, the system ensures that answers are grounded in verified data while fostering iterative

problem-solving processes. This design mitigates hallucinations, improves reasoning accuracy, and supports the seamless integration of new quantitative data into evolving experimental contexts.

Across its three development phases, LmRaC showcases innovations that redefine the integration of LLMs into scientific workflows. In the command-line phase, a two-pass RAG approach incorporates a "usefulness" metric to filter retrieved information, reducing hallucinations and enhancing precision. The web application phase introduce a fully interactive user interface to domain knowledge and experimental data and enhanced reasoning to enabling comprehensive, data-aware answers. Finally, the collaborative note-taking phase embeds case-based reasoning (CBR) into a unified platform, leveraging graph databases and vector embeddings for efficient knowledge management while promoting human-AI collaboration through iterative workflows. Together, these advancements create a dynamic, transparent, and adaptive system that amplifies human reasoning and facilitates collaborative scientific discovery.

The system's interdisciplinary design integrates human workflow principles with LLM-enhanced tools to enhance interaction and adaptability, providing a powerful tool for addressing complex, evolving problems in scientific research. By positioning LLMs as collaborative partners rather than autonomous oracles, this work lays the foundation for a new paradigm in AI-human collaboration, emphasizing transparency, ethical accountability, and dynamic knowledge integration. These contributions set the stage for more responsible and impactful applications of AI across critical domains, advancing human knowledge and decision-making.

AUTOBIOGRAPHICAL STATEMENT

Douglas Craig was born in Detroit, Michigan and grew up in its suburbs (Warren and Shelby Township). He first attended Oakland University where he earned a BSE in Computer Engineering, an MS in Computer Science & Engineering and completed coursework for his PhD in Systems Engineering. He restarted his PhD a few years later studying Cognitive Science and Philosophy (University of Michigan), but... He hoped to restart his PhD again in Translation Neuroscience, but... no. Meanwhile, he completed a BA in Philosophy along with a BS in Engineering Biology (both also from Oakland), then added an MFA in Creative Writing from Goddard College a couple years later (because he wanted to finish writing his novel).

Throughout his worklife and education there has been a common thread: a fascination with thinking machines. He was fortunate to learn how to program at the dawn of the home-computer revolution. He has been even more fortunate to be around to see large-language models and the promise of AI seemingly coming to fruition.

His doctoral studies at Wayne State University have, hopefully, added to the goal of thinking machines that benefit everyone. He has numerous publications on autonomous vehicles, protein design, oncology and artificial intelligence. This work has been cited over 1000 times, especially, and most gratifyingly, by researchers developing modern vaccines.

Outside of academia, Doug enjoys reading, writing, cooking, birding and hiking, sometimes all in the same day.

ProQuest Number: 30311314

INFORMATION TO ALL USERS

The quality and completeness of this reproduction is dependent on the quality
and completeness of the copy made available to ProQuest.



Distributed by

ProQuest LLC a part of Clarivate (2025).

Copyright of the Dissertation is held by the Author unless otherwise noted.

This work is protected against unauthorized copying under Title 17,
United States Code and other applicable copyright laws.

This work may be used in accordance with the terms of the Creative Commons license
or other rights statement, as indicated in the copyright statement or in the metadata
associated with this work. Unless otherwise specified in the copyright statement
or the metadata, all rights are reserved by the copyright holder.

ProQuest LLC
789 East Eisenhower Parkway
Ann Arbor, MI 48108 USA