Democratizing reliable knowledge-seeking with MyGPT: A Privacy-First, Open-Source Retrieval-Augmented Generation Platform

Jaimin Patel[1,2,†,*], Jude Downing[1,2,†], Hugh Ferguson[1,2], Thika You[1,2], Duccio Malinverni[1,2], Steve Mathew D. A.[1,2,8], Inês Chen[2], Madison Sluter[1,2], Beth Moorefield[1,2], Katalin Parej[2], Mukundan Ragavan[2], Cindy Morris[3], Dinesh Keerthi[3], Desiree Becerra Armada[4], Balint Meszaros[1,2], Vikas Trivedi[1,2], Shahinur Alam[5], Anthony Woodard[5], Dan Alford[5], Sagar Pathak[1,2], Cai Li[6], Renato Umeton[7], Carlos Rodriguez-Galindo[4], Catherine G. Lam[4], Stephen Gottschalk[3], Charalampos G. Kalodimos[2], and M. Madan Babu[1,2,7,*]

[1]*Center of Excellence for Data-Driven Discovery, Department of Structural Biology, St. Jude Children's Research Hospital, Memphis, TN, USA*

[2]*Department of Structural Biology, St. Jude Children's Research Hospital, Memphis, TN, USA*

[3]*Department of Bone Marrow Transplantation and Cellular Therapy, St. Jude Children's Research Hospital, Memphis, TN, USA*

[4]*Department of Global Pediatric Medicine, St. Jude Global, St. Jude Children's Research Hospital, Memphis, TN, USA*

[5]*Research Information Services, St. Jude Children's Research Hospital, Memphis, TN, USA.*

[6]*Biostatistics, St. Jude Children's Research Hospital, Memphis, TN, USA.*

[7]*Office of Data Science, St. Jude Children's Research Hospital, Memphis, TN, USA.*

[8]*College of Graduate Health Sciences, University of Tennessee Health Science Center, Memphis, TN, USA*

[†]*J.P. and J.D. contributed equally to designing the initial prototype for MyGPT.*

** Corresponding authors:* jaimin.patel@stjude.org, madan.babu@stjude.org

ABSTRACT

Large language models (LLMs) have transformed knowledge discovery, but their use in sensitive domains is limited by privacy concerns and risks of data leakage. Moreover, LLMs may hallucinate when queried about information outside their training data, undermining their reliability for scientific, clinical, or policy related tasks. We present MyGPT, a fully open-source, privacy-preserving retrieval-augmented-generation (RAG) platform that uses the latest open-source LLMs and operates locally or within secure institutional environments. MyGPT empowers users to query user-defined libraries such as biomedical literature, confidential clinical documents, and multi-lingual global health policies, while ensuring that sensitive data never leaves the secure environment. We introduce new confidence metrics, including "relevance scores" and "hallucination index," and provide explicit source citation from the library within a single interface to assess answer trustworthiness. MyGPT offers a cost-effective open-source alternative to commercial LLM services and democratizes knowledge-seeking across multiple domains, including use at remote facilities without internet access.

KEYWORDS

Large Language Models (LLMs), Generative AI, Generative Pre-trained Transformer (GPT), Artificial Intelligence, Retrieval-Augmented Generation (RAG), Hallucination Index (HI).

INTRODUCTION

Information seeking is fundamental to modern human activity. With the rise of Large Language Models (LLMs), users can now ask detailed questions that require synthesis, reasoning, and examination of multiple sources. However, LLMs have limitations that make them unreliable in areas requiring certainty, transparency, and reproducibility, such as scientific research[1,2]. They can generate plausible but incorrect responses, known as hallucination[2], due to insufficient human feedback, biases in the training dataset[3], and the exclusion of private and copyrighted material from training data. LLMs are also limited to data available up to their training cut-off date (e.g., December 2025 for GPT-5.5-Pro). Moreover, they may not perform well at tasks requiring synthesis of complex concepts for real-world, research-related purposes[1,4] such as literature review, where LLMs may provide biased answers or hallucinated sources that undermine their reliability and transparency[5] (**Supplementary Fig. 1**).

Use of commercial services such as ChatGPT and Gemini raises several important issues. First, they can collect user data, queries, and documents to train their AI/ML resources, posing a privacy concern[6]. While such services advise against sharing private, confidential, or sensitive information[4,7], users in academic, clinical, and public institutions may inadvertently leak sensitive data[8]. Consequently, institutions either restrict access or opt for costly enterprise versions[6]. Importantly, commercial services replace their older LLMs with newer versions, hindering reproducible research[8] (e.g., GPT-4o and GPT-4.1 were retired on Feb 2026). Moreover, institutes or scientists working in remote locations may be unable to use commercial services without reliable internet and cloud connections, further complicating access and security concerns[8].

LLM-powered products designed with a focus on privacy and customization may overcome these limitations[2,8,9]. Currently, there are two approaches by which this can be achieved. One approach is to fine-tune the LLM using domain-specific information or private data, where the LLM parameters are reweighted to learn field-specific or private information. However, this approach faces similar issues of trustworthiness and reliability. It also requires periodic fine-tuning of the LLM to incorporate new information and close monitoring of the fine-tuned version for privacy, data-leakage, LLM redundancy, and reproducibility. The other approach is the retrieval-augmented generation (RAG) concept, where LLMs trained on limited data can retrieve additional information from indexed documents to generate a response[10]. In simple terms, when the user asks a question, LLMs receive it with a set of instructions known as a 'prompt'. When additional supporting information is provided as 'context' along with the prompt, improved responses and reduced hallucinations have been observed[1]. Such relevant context can be retrieved from user-defined libraries of documents to ensure that they remain private and to increase response accuracy.

Providing exact sources, specific sections of the text within the sources, and confidence metrics for LLM-generated responses would be beneficial to address issues of transparency and trustworthiness and to identify instances of potential hallucinations[11]. Some commercial RAG solutions such as Microsoft Copilot, NotebookLM and ChatPDF provide source citing, but we are not aware of any that provide confidence metrics to identify potential hallucinations. In the past couple of years, rapid growth in Generative AI has enabled several open access RAG solutions for local installation, ranging from preliminary to advanced concepts. However, these solutions often depend on commercial API services, have limited document library sizes, and lack features (e.g., source surfacing, confidence metrics) to identify hallucinations.

To circumvent these limitations, we have developed MyGPT, an open-access, locally run, and customizable end-to-end RAG solution that enables researchers to ask questions from a curated library of documents or publications. MyGPT presents the exact text sources (e.g., paragraphs) to increase transparency and accuracy and provides confidence metrics that help identify instances of likely hallucination. MyGPT features a user-friendly interface to customize various components within the pipeline. This modularity provides an easy and future-proof way of swapping components[8] and allows users to upload documents

and interact in their preferred languages, minimizing language barriers to information access. In this way, MyGPT democratizes reliable knowledge-seeking by filling a gap in the current arsenal of LLM-based tools.

RESULTS

Outline of MyGPT architecture and workflow

We designed MyGPT as a software framework that (a) combines accurate and transparent context retrieval with source surfacing to validate LLM responses, (b) provides confidence metrics for validation, (c) incorporates a modular design to ensure future-proofing, and (d) is both domain- and language-agnostic. We also aimed to develop a software solution that creates a workflow with optimal RAG settings for a specific use case, enabling users to access a document library, validation tools, and a chat interface within a single browser window.

MyGPT architecture is composed of three sections: a user interface (UI), a backend server, and an LLM server (**Fig. 1**). The UI and backend server function to pre-process the library of documents to build a knowledge base and to perform real-time question-and-answer (Q&A) by providing relevant context using context engineering, while the LLM server is key for response generation.

In the pre-processing step, the UI first receives a library of documents from the user via either a reference management tool (e.g., Zotero[12]; **Methods**) or direct upload. The backend server divides the documents into smaller chunks. Chunking is essential for effective information processing, as LLMs have a limited context window size and currently cannot accept entire or large documents. The chunks are then stored in two places: a BM25[13] (Best Matching 25)-tokenized library for keyword search and an embedding-based vector library within a vector database for semantic search.

For semantic search, the chunks are converted into embeddings (numerical vectors) that capture similarity and thus acquire specific meanings (semantics) defined by the embedding model. Different embedding models, trained on domain-specific data, can enhance the response of an LLM by capturing specific relationships between words and sentences in that particular domain[14]. The vectorized representations of knowledge sources are then stored in a searchable database hosted through the backend (**Fig. 1a**). The embedding models organize semantically similar terms close to each other in the vector space[14]. This concept also applies to sentences and paragraphs, where entities with similar meanings are positioned closer to one another, enabling better performance in similarity searches[15], as illustrated in **Supplementary Fig. 2**. Similarly, for keyword search, the same chunks are tokenized and indexed using the BM25 algorithm during pre-processing. In our real-world use cases described below, domain-specific private documents were never used to train an embedding model or LLM. BM25 ensures that queries with domain-specific terms match relevant chunks of documents when semantic search fails.

After pre-processing, the UI displays both the library as a list of documents and a chat box for real-time Q&A (**Fig. 2**), and users can upload additional documents and customize the various components of MyGPT (see below). When a user asks a question, it is sent to the backend server to retrieve the relevant context. For semantic search, the query is converted into a vector embedding by the same embedding model used to preprocess the library and passed to the vector database (**Fig. 1b**). The vector database's similarity search algorithm compares the question to the chunks from the library and identifies the most similar chunks. For keyword search, the query is submitted to the BM25 algorithm, which identifies and returns the chunks with the highest BM25 scores. Chunks generated from both approaches are then evaluated by a Reranker (such as cross-encoders[16] developed by Sentence Transformers) to assess their relevance in addressing the query. Only those chunks that meet the Reranker threshold are retained as the final k chunks after duplicate entries have been consolidated. These k chunks are then provided to the LLM server as the 'golden context' for the query, enhancing the LLM response[17]. The LLM, which runs locally on a personal computer or a secure server/cloud infrastructure, performs context-based Q&A, generates a

response, and sends it back to the UI (**Fig. 2**). The UI then displays the source material (i.e., chunks from specific documents) used as context to generate the response and precisely specifies their location (i.e., as document pages with highlighted sentences or paragraphs). The UI additionally displays confidence metrics calculated by the backend server that can help identify instances of hallucinations (**Fig. 2;** see below and **Methods**).

MyGPT workflow customization options

The individual sections within MyGPT are comprised of modularized components (**Fig. 3**) that can be replaced or customized through the UI to provide a seamless user experience (**Supplementary Fig. 3**):

- *Chunking:* MyGPT offers two chunking methods: fixed-size or document-structure-preserving recursive chunking (**Methods**). While both methods allow users to adjust chunk overlap (i.e., the number of characters shared between two consecutive chunks[18,19]), structure-preserving chunking additionally enables users to focus on a specific section within a document or on common sections across all documents (e.g. abstract or results for scientific publications).

- *Embedding models:* MyGPT users can deploy any open-source embedding model available from Hugging Face (a platform to host and share open-source AI/ML resources) or via Ollama (an LLM hosting tool). Over 400 different[20,21] embedding models are available, including general-purpose embedding models such as nomic-embed-text, or a domain-specific embedding model, such as MedCPT-query-encoder (trained on query-article pairs from PubMed search logs) for the biomedical domain.

- *Keyword search (BM25):* MyGPT uses the BM25 algorithm[13,22] to help queries with domain-specific terms find relevant document sections. BM25 is enabled by default for English, but users can select another language or skip this step as needed.

- *Reranker:* To validate how relevant each chunk is to answer a user's question, MyGPT uses a QNLI (Question Natural Language Inference) cross-encoder[16] as a Reranker. This Reranker has been trained on Wikipedia passages paired with corresponding answerable questions. It provides a ranked score for each chunk based on its ability to answer the user's query. Based on the score, the chunks can be sorted from the best to the worst. While it was originally trained on English, users can select different or multilingual Rerankers depending on their needs. Once the LLM generates an answer, MyGPT determines the textual entailment of chunks using a different Reranker trained on an NLI dataset. This Reranker predicts whether two sentences (LLM answer and the chunk) contradict one another, entail one another, or are neutral. Based on these predictions, MyGPT calculates the Answer Relevance Score (ARS) and the Hallucination Index (HI), in addition to BM25 scores and vector distances.

- *Vector space distance function* (i.e., semantic similarity search metric): To identify relevant chunks, MyGPT users can choose from three distance functions: Euclidian distance, cosine similarity, and Inner product (**Methods**).

- *Large language models (LLMs):* MyGPT provides a UI for downloading several popular open-source LLMs via Ollama or any of the additional 100+ open-source LLMs using a command-line interface, enabling their local deployment and circumventing the need to connect to proprietary online services[8,23]. Ollama's performance depends on the available GPU, making it an essential component for running MyGPT. We performed comprehensive benchmarking to evaluate several LLMs in terms of ideal hosting environment, minimum requirements, average speed of response to a user query, and cost estimates of hosting in a cloud infrastructure (**Supplementary Tables 1–3)**. For example,

relatively small Llama3:8b and Llama2:7b[24] (~8 billion and ~7 billion parameters, respectively) can run on a machine with a single GPU and generate answers quickly. In contrast, a large LLM such as Llama3:70b (~70 billion) must be hosted on a machine with a powerful GPU for optimal performance but can generate more detailed answers. Users can also interact directly with the LLMs by turning off the RAG pipeline through the UI.

- *Creativity vs precision and prompting:* LLM models generate text by predicting the next words in the sequence, and this can be customized by adjusting specific LLM parameters: top-K (the most probable words to consider for the prediction), top-P (the threshold of probabilities of words), and temperature (the randomness of words selected by top-P and top-K). In this way, users can control the creativity or precision of the responses generated by the LLM. Users can also limit the output from the LLM by changing the default prompt through the UI, e.g. answer in a single word (yes/no), to restrict creativity.

- *Model context protocol (MCP)*: This open standard developed by Anthropic[25] facilitates integration of LLM applications with external public or private tools and data. MyGPT offers a basic implementation of the MCP server and client, enabling its integration with other private LLM applications and data sources hosted within the same AI ecosystem (**Supplementary Information)**.

Confidence metrics

In RAG applications, the risk of hallucinations increases if relevant information is missing from the documents; users must therefore evaluate the answers provided by the LLM and verify their sources to ensure accuracy. To address this challenge, we developed and implemented 'relevance scores' both for user queries and MyGPT-generated answers, as well as a 'hallucination index'. MyGPT provides built-in functionality to calculate these metrics, which can be further optimized for specific use cases. We define these metrics below:

- *Question Relevance Score (QRS)*: provided when the user submits the question, it serves as a metric to assess whether the user-provided library may have the necessary information to respond correctly to the question asked. To calculate Question Relevance Score (QRS), we define QRS as

$$QRS = \left(a \times \frac{\sum_{i=1}^{k} Q_{sem,i}}{k}\right) + \left(b \times \frac{\sum_{i=1}^{k} Q_{key,i}}{k}\right) + \left(c \times \frac{\sum_{i=1}^{k} Q_{rerank,i}}{k}\right) \quad (1)$$

where $Q_{sem,i}$ is the Semantic score, $Q_{key,i}$ is the Keyword score and $Q_{rerank,i}$ is the Rerank score for the i^{th} chunk, and a, b and c are weighting factors for the 3 contributions (domain-specific a,b,c can be determined by global sensitivity analysis of a reference curated Q&A dataset, see **Methods;** Finding the optimal settings for QRS, ARS, and HI). k denotes the number closest retrieved chunks after applying thresholds from semantic search, keyword search and reranking (see **Methods**).

The semantic score $Q_{sem,i}$ for chunk i is given by,

$$Q_{sem,i} = 1 - \frac{QCsem_i - QCsem_{best}}{QCsem_{worst} - QCsem_{best}} \quad (2)$$

where $QCsem,i$ is the distance between the embeddings of the question and the of chunk i in the vector database. The normalized distance is subtracted from 1 because lower distances correlate to higher semantic similarities. $QCsem_{best}$ and $QCsem_{worst}$ are the empirical highest and lowest QC distances that were observed for 627 questions and retrieved context chunks for the benchmark dataset (details in **Supplementary Data**). Both vary for different embedding models. The method to calculate these values as well as precalculated values for five populat embedding models are

provided for the user. We have calculated $QCsem_{best}$ and $QCsem_{worst}$ for several embedding models and provide resources to calculate them for other embedding models (see **Methods**).

The Keyword score $Q_{key,i}$ similarly defined as

$$Q_{key,i} = \frac{QCkey_i - QCkey_{best}}{QCkey_{worst} - QCkey_{best}} \quad (3)$$

where $QCkey,i$ is the score for the question and the i^{th} chunk calculated by the BM25 algorithm. $QCkey_{best}$ and $QCkey_{worst}$ are the empirical highest and lowest $QCkey$ BM25 scores that were observed for 627 questions and retrieved context chunks for the benchmark dataset (details in **Supplementary Data**).

The rerank score $Q_{rerank,i}$ for chunk *i* (described above) is directly generated by the Reranker model and range from 0 to1. The cross-encoder library provides an option to use the sigmoid function to produce min-max-normalized scores between 0 and 1.

- *Answer Relevance Score (ARS):* measures the extent to which MyGPT used the context (i.e., source material) retrieved from the documents to generate the answer. ARS is defined as

$$ARS = \left(x \times \frac{\sum_{i=1}^{k} A_{sem,i}}{k}\right) + \left(y \times \frac{\sum_{i=1}^{k} A_{key,i}}{k}\right) + \left(z \times \frac{\sum_{i=1}^{k} A_{entail,i}}{k}\right) \quad (4)$$

where $A_{sem,i}$ is the Semantic score, $A_{key,i}$ is the Keyword score and $A_{entail,i}$ is the Textual entailment score using the Reranker for the i^{th} source. x,*y,z* are weighting factors for the 3 contributions (domain-specific values can be determined by global sensitivity analysis of a reference curated Q&A dataset, see **Methods**)

Similarly to $Q_{sem,i,}$ Semantic score $A_{sem,i}$ is defined as,

$$A_{sem,i} = 1 - \frac{ACsem_i - ACsem_{best}}{ACsem_{worst} - ACsem_{best}} \quad (5)$$

where $ACsem_i$ is the distance between the embeddings of the question and the i^{th} chunk in the vector database. We subtract the normalized distance from 1 because the lower the distance, the better the semantic similarity. $ACsem_{best}$ and $ACsem_{worst}$ are the empirical highest and lowest AC distances that were observed for 627 answers and retrieved context chunks for the benchmark dataset (details in **Supplementary Data**). Both vary for different embedding models. The method to calculate these values as well as precalculated values for five populat embedding models are provided for the user.

Similarly, the Keyword score $A_{key,i}$ is defined as

$$A_{key,i} = \frac{ACkey_i - ACkey_{best}}{ACkey_{worst} - ACkey_{best}} \quad (6)$$

where $ACkey_i$ is the score for the answer and the i^{th} chunk calculated by the BM25 algorithm. $ACkey_{best}$ and $ACkey_{worst}$ are empirical highest and lowest $ACkey$ that were observed for 627 answers and retrieved context chunks for the benchmark dataset (details in **Supplementary Data**).

We calculated A_{entail} using a Reranker that provides a textual entailment label for each of the retrieved chunks and the LLM generated answer. This entailment label is either ‘entailment’, ‘contradiction’ or

'neutral'. To calculate the A_{entail} score, we converted the label to a score for each chunk, either 1, -1, or 0.5 values for entailment, contradiction, and neutral, respectively, and averaged these values across the k chunks.

- *Hallucination Index (HI):* is a weighted linear combination of QRS and ARS. This score helps users identify likely instances of hallucination and serves as a confidence metric for the accuracy and reliability of the generated answers. In this way, MyGPT can flag likely instances of hallucinations, either because the library is less relevant to the question or because the context retrieved from the document lacks sufficient information for the LLM to generate an accurate response. We defined HI by combining QRS and ARS, given by

$$HI = 1 - \left(\frac{p \times QRS}{p+q}\right) - \left(\frac{q \times ARS}{p+q}\right) \quad (7)$$

where p and q are weighting factors that can be determined by global sensitivity analysis on a reference Q&A dataset (see **Methods**). HI is normalized to [0,1], such that when QRS and ARS are 1, HI will be zero (and conversely). HI values closer to 1 (or 100%) indicates likely hallucination and values closer to 0 (0%) indicates less hallucination and more certainty in the response.

In addition to ensuring that the implemented RAG concept works as expected (**Supplementary Information**), we assessed the broader utility of our RAG solution by comparing the performance of MYGPT against other RAG solutions. Specifically, we benchmarked MyGPT and its confidence metrics using three publicly available benchmarking datasets as described below.

Benchmarking MyGPT and other open-source RAG solutions for public datasets

We directly compared the performance of MyGPT with that of four open-source RAG solutions: RAGFlow, Dify, PrivateGPT, and Kotaemon, using three public evaluation datasets: PubMedQA[26], BioASQ[27], and Open-rag-bench. To do so, we collected 294 questions (294 documents) from PubMedQA (a biomedical question-answering dataset collected from PubMed abstracts), 163 questions (397 documents) from BioASQ (a manually curated evaluation dataset on large-scale biomedical semantic indexing and question answering), and 170 questions (165 documents) from the Open RAG Benchmark datasets designed to evaluate RAG systems using arXiv PDFs (**Methods**).

MyGPT achieved 81.6%, 91.4%, and 92.9% accuracy on PubMedQA, BioASQ, and Open RAG Benchmark, respectively (**Fig. 4a**), outperforming RAGFlow (77.5%, 90.2% and 90% across the 3 datasets), Dify (68%, 65.6% and 78.8%), PrivateGPT (51%, 74.8% and 78.2%) and Kotaemon (45.6%, 76.1% and 55.8%) in both accuracy, document processing, and source highlighting. The confidence metrics (QRS, ARS, and HI) obtained using MyGPT differentiated between correct and incorrect answers and conclusively identified instances of hallucination (**Fig. 4b-d**). We observed that both QRS (p = 4x10^{-3}; effect size = 0.11 (biserial rank correlation r, small); Mann Whitney U test) and ARS (p = 1x10^{-5}; effect size = 0.17 (r, small); Mann Whitney U test) were significantly higher for relevant questions and correct answers, and HI (p = 1x10^{-5}; effect size = 0.17 (r, small); Mann Whitney U test) was significantly higher for irrelevant questions and incorrect answers. These findings demonstrate that MyGPT (and the confidence metrics) works as anticipated and that MyGPT outperforms the open-source RAG solutions RAGFlow, Dify, PrivateGPT, and Kotaemon by generating more correct answers in our benchmark.

There is growing evidence and concern that LLMs that are trained on public data, including research articles[28], making it challenging to determine which specific publications were included in the training datasets. To address this concern, rather than evaluating the LLM as an independent component, we evaluated the entire RAG pipeline of MyGPT with other open-source RAG platforms with the same LLM. Due to our limited access to pdf versions of published articles and the manual effort required for validation,

we used subsets of these datasets. However, power calculations were used to guide and interpret the benchmarking sample sizes (**Methods**), and both the number of Q&A pairs and the document count reflect those considerations.

Real-world use cases and performance of MyGPT using public and confidential, private documents
To highlight real world applications of MyGPT in the different domains, we describe three examples of use cases in clinical sciences, basic sciences, and global health at St. Jude Children's Research Hospital as below:

Private clinical documents: MyGPT can be securely deployed within hosting environments that require private and confidential protocols, such as those maintained within hospitals or clinical settings. For example, we implemented MyGPT for the Department of Bone Marrow Transplantation and Cellular Therapy (BMTCT) at St. Jude Children's Research Hospital (**Methods**) to allow staff to easily access and query Standard Operating Procedures (SOP) documents. Doctors and nurses used MyGPT to query specific information from a library of SOPs (**Figure 5**) and verified the MyGPT responses against the original documents for validating the responses. Since SOPs are meticulously prepared and approved through a rigorous process at St. Jude, responses generated by the language model should be accurate, institution-specific, and verifiable with source citations. MyGPT showed 94.85% accuracy in answering the questions for the evaluation dataset we prepared for this use case, with significantly higher relevance scores for correct answers than for incorrect answers and a lower hallucination index (p = $1.4x10^{-4}$; effect size = 0.33 (r, medium); Mann Whitney U test). MyGPT may also be utilized for other confidential clinical documents, such as patient reports and Electronic Health Records (EHRs).

Biomedical literature: MyGPT is useful for querying biomedical literature, particularly for exploring new literature that is not included in LLM training or for querying bespoke libraries of published articles collected by research groups. MyGPT was implemented in the BlueSky Kinase project of the Structural Biology department at St. Jude to test its ability to retrieve technical and highly specific details of biochemical assays and expression systems that monitor substrate phosphorylation and kinase activity and to determine whether these effects were referenced in any figure or table descriptions. To validate its performance, we compiled 83 biomedical publications for 26 different protein kinases and created 282 Q&A pairs with input from experts in the field (principal and senior scientists working on kinases). The answers produced by MyGPT were reviewed by a biocurator, who reported 73% (206/282) accuracy. When we repeated the experiment with 15 chunks per question rather than the default value of 5 chunks, we obtained 21 more correct answers with 80.5% (227/282) accuracy with the same customization showing significant improvement (p = $0.7x10^{-2}$; effect size = 0.34 (phi coefficient Φ, medium); McNemar chi-square test). We also repeated the experiment with structure-preserving chunking and 5 chunks, obtaining 75.8% (214/282) accuracy, which did not provide a significant improvement (p = 0.27; effect size=0.15 (Φ, small); McNemar chi-square test) over fixed chunking. Although structure-preserving chunking does not provide improvement here, it should be useful when the user wants to focus on a specific section or avoid specific sections while generating a response. For publications containing the requisite information, MyGPT provided highly relevant answers with minimal hallucination, with exact paragraphs highlighted and a significantly lower hallucination index for correct than for incorrect answers (**Supplementary Fig. 4; Methods**). Moreover, the MyGPT pipeline completed the task in less than 2 hours, a process that would have taken a biocurator approximately 30-40 hours to finish (about 25-30 minutes per article) by reading and extracting data from 83 articles. Thus, MyGPT's confidence metrics and source-surfacing capability enable researchers to assess the trustworthiness and accuracy of LLM-generated answers while keeping the knowledge base constrained to a selected library or even to a single document.

Multi-lingual global health policies: MyGPT allows users to interact with documents in various languages without concerns about language compatibility. This is possible due to its modular design, which enables users to switch between different embedding models and BM25 algorithms, including those trained on

multiple languages, such as bge-m3, Paraphrase-Multilingual, and nomic-embed-text-v2-moe. When a user uploads a document in a language other than English, the embedding model can categorize the content alongside other chunks with similar meanings, irrespective of the original language. Although a user may not be familiar with the context in an unfamiliar language, the language model used to generate the final answer can recognize and translate it to the language in which the user asks their question (**Methods**). This can be especially beneficial where rapid and accurate data extraction and interpretation are warranted from length and compled document sets such as national health policies. Such documents guide critical decision making and funding, and yet are often laden with jargon, interdependencies across sections, and where interpretation is highly dependent on country, context and subtext. We leveraged this capability to implement MyGPT with the department of Global Pediatric Medicine, which has developed a Policy Monitor with national health policy documents from 197 countries and territories around the world, in 46 different languages. To validate the use case, we compiled 20 health policies: 10 each for Spanish and French. We comipled 16 questions to evauate each policy across five topics commonly queried by ministry teams and counterparts in national policy planning. These include topics in health and non-communicable diseases in children, pediatric and adolescent cancer types, existence of a staffed national cancer program, access to vaccination to prevent cervical cancer in adults and access to essential imaging (**Methods** and **Supplementary Data**). The answers produced by MyGPT were independently reviewed by an expert team of health policy analysts with the department of St. Jude Global Pediatric Medicine using structured rubrics and tools, and found to be 80% (128/160) accurate for Spanish and 88% (141/160) for French (**Supplementary Fig. 5**).

Machine learning-based optimization of hallucination index

As we observed significantly higher HI scores for incorrectly answered questions in the 3 real-world examples used to assess MyGPT, we asked whether the information contained in ARS/QRS could be used to train a machine learning classifier to explicitly predict whether answers are likely to be correct or incorrect. To achieve this, we trained a random forest algorithm using 20 input features derived from QRS and ARS calculations (**Methods**). We observed 87.35% (±0.4%) train and 74.17% (±4.16%) test ROC-AUC (**Supplementary Fig. 6**) for a random forest algorithm on 1045 Q&A, combining public and private datasets (80:20 train-test split; 10 replicates; **Methods**). We also observed that this optimized HI score showed marked improvement in separating correct from incorrect responses for 3 public datasets ($p = 2.2\times10^{-16}$; effect size = 0.38 (r, medium); Mann Whitney U test) and private datasets: Bone Marrow Transplantation SOPs ($p = 1.4\times10^{-4}$; effect size = 0.33 (r, medium); Mann Whitney) and Protein Kinases ($p = 8.8\times10^{-16}$; effect size = 0.59 (r, large); Mann Whitney). While we present the Random Forest model as the default to predict HI, users can either retrain the model using the scripts that we provide and replace it in the source code or can train their own model with their own custom settings and libraries to obtain optimized HI scores. The latter option requires generating a document library and at least 130 domain-specific questions and answers, each with correct vs. incorrect labels (see **Methods**).

LLM hallucination can be independently evaluated by other methods such as SelfCheckGPT[29] and semantic entropy[30] through generation of multiple responses. However, these are not well-suited for RAG contexts, as SelfCheckGPT and semantic entropy both require substantial GPU resources, and SelfCheckGPT is ineffective for smaller LLMs such as GPT-3 and Llama[29]. Moreover, if the correct context is provided, even smaller LLMs can produce accurate answers. Thus, by providing QRS, ARS and HI metrics and explicit source citing, MyGPT offers a comprehensive and generalizable method to quantify the correctness of LLM-generated answers, thereby increasing the reliability and transparency of RAG applications. We anticipate that MyGPT can be used across range of environments, including routine tasks in hospitals, human resources, administration, legal fields, and commercial contexts or by students and the general public, providing a comprehensive, end-to-end, seamless user experience from a single browser window (**Figure 6**).

Considerations and limitations

MyGPT uses the textual content from documents to create a vector database, and it is currently unable to interpret tabular data, figures, mathematical equations, formulas, or chemical structures. This limitation can be addressed in the future with multimodal LLMs (e.g., text + images using Gemme3; text + audio using Gemma4). The open-source nature of MyGPT makes such future extensions easily implementable by the community. Currently, the user needs to provide the documents to create the library, as MyGPT does not automatically retrieve all relevant publications on a topic of interest; if supporting or contradicting information is absent from the user-provided library and/or was not included in the LLM training data, MyGPT could generate a biased response by citing only the available source in the library.

While ARS, QRS, and HI metrics can help identify instances of hallucination, they do not eliminate them and may need to be optimized to distinguish between correct and incorrect responses for specific libraries and scientific domains. These metrics rely on the properties of user input, such as question type and clarity and on MyGPT parameters such as the embedding model used. In addition, users need to install MyGPT on an individual device for personal use and on Edge AI devices, a server, or a secure cloud for institutional use, requiring maintenance as an internal web resource. The size of the library can influence the number of chunks per library and impact the uploading time, but it does not affect the duration of different steps in real-time Q&A once uploading is complete (**Supplementary Table 4)**. In our ongoing efforts, we also observed that advanced semantic chunking, which is computationally expensive, does not provide a substantial performance improvement[31]. Despite these considerations, local installation capabilities allow users to customize the pipeline according to their requirements, to securely deploy it within a firewall, eliminates the need to send confidential information to third parties, minimizes data leakage, and makes MyGPT more affordable than commercial cloud-hosted LLM products for individuals, research groups, and organizations.

DISCUSSION

MyGPT is an open-source, easy to install, portable, and customizable RAG platform that provides confidence metrics based on context engineering and a machine learning algorithm to infer correctness of LLM-generated answers. MyGPT's implementation of RAG concepts consistently delivers above 80% accuracy across public and private datasets, and its unique modular framework enables users to both integrate the latest LLMs and embedding models and to perform multilingual Q&A in their preferred language, thereby minimizing language barriers to information access. Users can search their private, curated document library and obtain reliable answers from multiple documents simultaneously. Thus, MyGPT fills a crucial gap in knowledge discovery tools by enabling users to efficiently extract verified insights from multiple private documents.

For all public and private datasets, we evaluated MyGPT with various customizations and selected the best-performing setting (**Methods**). To identify the optimal combination of MyGPT components for a user's own library or use case (LLMs, embedding model, distance function), we suggest selecting top-performing LLMs from leaderboards such as Chatbot Arena[32,33] and embedding models from the Massive Text Embedding Benchmark (MTEB) Leaderboard[20,21] recommended by sentence transformer studies[34]. For other customizations, domain-specific LLMs or embedding models beyond the leaderboards, users can also utilize automated evaluation frameworks such as Retrieval-Augmented Generation Assessment (Ragas)[35,36] with MyGPT APIs.

We note that MyGPT includes solutions similar to those offered by commercial products such as ChatPDF and NotebookLM or by open-source products such as PrivateGPT, Kotaemon, RAGFlow, Dify, Verba, LlamaIndex, and BioChatter. However, in contrast to those products, MyGPT is a complete, end-to-end solution built with open-source tools that are small enough for local installation on a laptop/personal computer, Edge AI devices, institutional servers/VMs, or the cloud. MyGPT can be installed and initiated by a script orchestrating prebuilt Docker images, eliminating complex debugging and environment

dependencies for less tech-savvy users. When compared to end-to-end solutions such as PrivateGPT, Kotaemon, RAGFlow, and Dify, MyGPT offers greater accuracy and broader range of features, including confidence metrics, library management, and document highlighting (**Figure 6b)**.

MyGPT also offers users complete control over their data. In research, healthcare, and industry, a substantial amount of knowledge is locked in private documents, necessitating a solution for a contextual, retrievable, and scalable knowledge base. Future RAG development with agentic architecture must access this private knowledge based on user permissions. Agentic workflows can effectively integrate MyGPT's' confidence metrics to filter out low-relevance references and to rephrase less relevant questions, thereby enhancing decision-making and improving accuracy. Moreover, the parameters collected during each Q&A session are used by MyGPT to train a low-latency machine learning model to improve confidence metrics, which can be further refined based on the types of questions in different domains. Current commercial Deep Research applications either do not support private documents or they require a subscription to commercial services. In contrast, MyGPT offers an open-source API with built-in user access control that enhances agentic workflows, as well as Model Context Protocol (MCP) to enhance ongoing efforts such as Open Deep Research[37] and multi-agent systems for healthcare (MASH)[38].

With its ability to run on Edge AI devices such as Nvidia Orin Nano (**Supplementary Fig. 7**) and to handle documents offline, MyGPT provides a cost-effective alternative to commercial LLM services, democratizing access to reliable knowledge-seeking tasks, particularly in remote medical and research facilities without internet access. As vector embeddings for specific libraries are independent of GPT models, sharing pre-made MyGPT libraries with the community may accelerate and broaden opportunities for knowledge seeking and scientific discovery. This capability empowers researchers to use publications from open research initiatives such as PubMed Central[39] or openRxiv[40], to create private and open-source instances of AI-powered literature research tools such as Consensus[41] or essential RAG-component of specialized LLM-based agentic tools such as CRISPR-GPT[42] and PrimeGen[43]. As we look to the future of generative AI, personalized GPT products are increasingly likely to become a reality. An open-source, privacy-focused retrieval-augmented generation framework presents a compelling path for generative AI that balances accessibility, accountability, and user autonomy.

REFERENCES

1 Cascella, M., Montomoli, J., Bellini, V. & Bignami, E. Evaluating the Feasibility of ChatGPT in Healthcare: An Analysis of Multiple Clinical and Research Scenarios. *J Med Syst* **47**, 33 (2023). https://doi.org/10.1007/s10916-023-01925-4

2 Eloundou, T., Manning, S., Mishkin, P. & Rock, D. GPTs are GPTs: An Early Look at the Labor Market Impact Potential of Large Language Models. *ArXiv* **abs/2303.10130** (2023).

3 Alkaissi, H. & McFarlane, S. I. Artificial Hallucinations in ChatGPT: Implications in Scientific Writing. *Cureus* **15**, e35179 (2023). https://doi.org/10.7759/cureus.35179

4 OpenAI. *How does OpenAI use my personal data?*, https://openai.com/policies/row-privacy-policy/ (2024).

5 Kwong, J. C. C., Wang, S. C. Y., Nickel, G. C., Cacciamani, G. E. & Kvedar, J. C. The long but necessary road to responsible use of large language models in healthcare research. *NPJ Digit Med* **7**, 177 (2024). https://doi.org/10.1038/s41746-024-01180-y

6 Lin, Z. Techniques for supercharging academic writing with generative AI. *Nat Biomed Eng* **9**, 426–431 (2025). https://doi.org/10.1038/s41551-024-01185-8

7 Gemini. *Gemini Apps Privacy Notice*, https://support.google.com/gemini/answer/13594961?visit_id=638586578410247935-92666435&p=privacy_notice&rd=1#privacy_notice (2024).

8 Hutson, M. Forget ChatGPT: why researchers now run small AIs on their laptops. *Nature, Technology Feature* (2024).

9 Xu, R., Feng, Y. & Chen, H. ChatGPT vs. Google: A Comparative Study of Search Performance and User Experience. *ArXiv* **abs/2307.01135** (2023).

10 Lewis, P. *et al.* Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *ArXiv* **abs/2005.11401** (2020).
11 Nowogrodzki, J. ChatGPT for science: how to talk to your data. *Nature* **631**, 924–925 (2024). https://doi.org/10.1038/d41586-024-02386-6
12 Zotero.org. https://www.zotero.org/
13 Kamphuis, C., de Vries, A. P., Boytsov, L. & Lin, J. Which BM25 Do You Mean? A Large-Scale Reproducibility Study of Scoring Variants. *Advances in Information Retrieval, Ecir 2020, Pt Ii* **12036**, 28–34 (2020). https://doi.org/10.1007/978-3-030-45442-5_4
14 AWS. *What are embeddings in machine learning?*, https://aws.amazon.com/what-is/embeddings-in-machine-learning/
15 Jing, Z. *et al.* When Large Language Models Meet Vector Databases: A Survey. *arXiv preprint arXiv:2402.01763* (2024).
16 Reimers, N. & Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (Emnlp-Ijcnlp 2019)*, 3982–3992 (2019).
17 Omar, M., Ullanat, V., Loda, M., Marchionni, L. & Umeton, R. ChatGPT for digital pathology research. *The Lancet Digital Health* **6**, e595–e600 (2024). https://doi.org/10.1016/S2589-7500(24)00114-6
18 Peter, A. *What Chunk Size and Chunk Overlap Should You Use?*, https://dev.to/peterabel/what-chunk-size-and-chunk-overlap-should-you-use-4338 (2023).
19 Hoang, L. *How Chunk Sizes Affect Semantic Retrieval Results*, https://ai.plainenglish.io/investigating-chunk-size-on-semantic-results-b465867d8ca1
20 Muennighoff, N., Tazi, N., Magne, L. & Reimers, N. MTEB: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316* (2022).
21 *Massive Text Embedding Benchmark (MTEB) Leaderboard*, https://huggingface.co/spaces/mteb/leaderboard
22 Lù, X. H. BM25S: Orders of magnitude faster lexical search via eager sparse scoring. *arXiv* (2024). https://doi.org/https://doi.org/10.48550/arXiv.2407.03618
23 Jeffery Morgan, M. C. *Ollama*, https://ollama.com/ (2021).
24 Touvron, H. *et al.* Llama 2: Open Foundation and Fine-Tuned Chat Models. *ArXiv* **abs/2307.09288** (2023).
25 Model Context Protocol (MCP) (GitHub https://github.com/modelcontextprotocol, 2024).
26 Jin, Q., Dhingra, B., Liu, Z. P., Cohen, W. W. & Lu, X. H. PubMedQA: A Dataset for Biomedical Research Question Answering. *2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (Emnlp-Ijcnlp 2019)*, 2567–2577 (2019).
27 Balikas, G., Krithara, A., Partalas, I. & Paliouras, G. BioASQ: A Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering. *Lect Notes Comput Sc* **9059**, 26–39 (2015). https://doi.org/10.1007/978-3-319-24471-6_3
28 Gibney, E. Has your paper been used to train an AI model? Almost certainly. *Nature* **632**, 715–716 (2024). https://doi.org//10.1038/d41586-024-02599-9
29 Manakul, P., Liusie, A. & Gales, M. J. F. SELFCHECKGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. *2023 Conference on Empirical Methods in Natural Language Processing (Emnlp 2023)*, 9004–9017 (2023).
30 Farquhar, S., Kossen, J., Kuhn, L. & Gal, Y. Detecting hallucinations in large language models using semantic entropy. *Nature* **630** (2024). https://doi.org/10.1038/s41586-024-07421-0
31 Qu, R., Tu, R. & Bao, F. S. Is Semantic Chunking Worth the Computational Cost? Findings of the Association for Computational Linguistics: NAACL 2025, pages 2155–2177, Albuquerque, New Mexico. Association for Computational Linguistics.
32 *LMSYS Chatbot Arena Leaderboard*, https://lmarena.ai/?leaderboard
33 Chiang, W.-L. *et al.* Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132* (2024).
34 *Pretrained Models* https://www.sbert.net/docs/sentence_transformer/pretrained_models.html
35 *Ragas (RAG ASsement)* https://docs.ragas.io/en/stable/index.html
36 Es, S., James, J., Espinosa-Anke, L. & Schockaert, S. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217* (2023).
37 Open Deep Research (GitHub https://github.com/langchain-ai/open_deep_research).

38 Moritz, M., Topol, E. & Rajpurkar, P. Coordinated AI agents for advancing healthcare. *Nat Biomed Eng* **9**, 432–438 (2025). https://doi.org/10.1038/s41551-025-01363-2
39 Central, P. *PubMed Central*, https://pmc.ncbi.nlm.nih.gov/ (2000).
40 OpenRxiv. *OpenRxiv*, https://openrxiv.org/ (2025).
41 consensus. *Consensus - AI-powered search engine for scientific research papers*, https://consensus.app/home/about-us/ (2021).
42 Qu, Y. *et al.* CRISPR-GPT for agentic automation of gene-editing experiments. *Nat Biomed Eng* (2025). https://doi.org/10.1038/s41551-025-01463-z
43 Wang, Y. *et al.* Accelerating primer design for amplicon sequencing using large language model-powered agents. *Nat Biomed Eng* (2025). https://doi.org/10.1038/s41551-025-01455-z
44 Trychroma.com. https://www.trychroma.com/
45 Lopez, P. GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction for Scholarship Publications. *Research and Advanced Technology for Digital Libraries, Proceedings* **5714**, 473–474 (2009).
46 Breiman, L. Random forests. *Mach Learn* **45**, 5–32 (2001). https://doi.org/Doi 10.1023/A:1010933404324
47 Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J Mach Learn Res* **12**, 2825–2830 (2011).
48 Lundberg, S. M. & Lee, S. I. A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems 30 (Nips 2017)* **30** (2017).

FIGURES

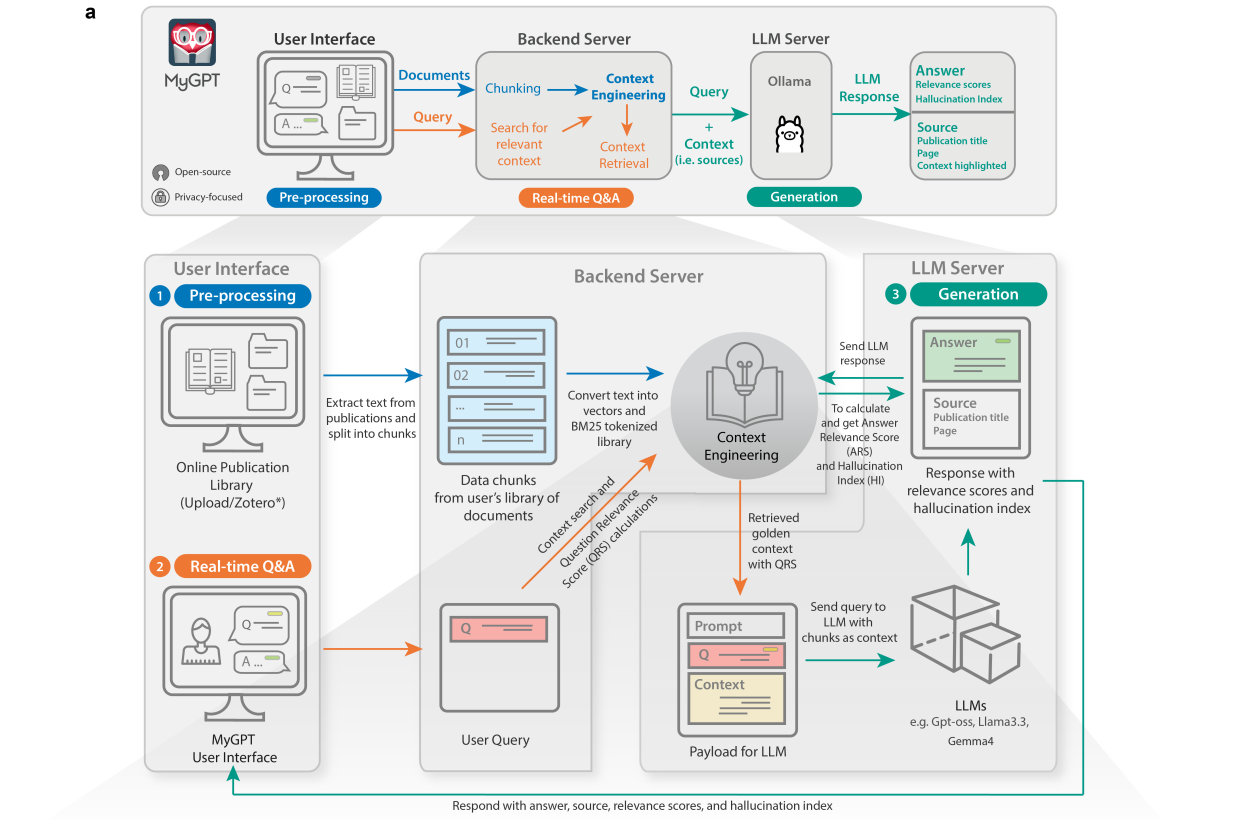

a
MyGPT
Open-source
Privacy-focused
User Interface
Documents
Query
Pre-processing
Backend Server
Chunking
Context Engineering
Search for relevant context
Context Retrieval
Real-time Q&A
LLM Server
Ollama
Query + Context (i.e. sources)
Generation
LLM Response
Answer
Relevance scores
Hallucination Index
Source
Publication title
Page
Context highlighted
User Interface
1 Pre-processing
Online Publication Library (Upload/Zotero*)
2 Real-time Q&A
MyGPT User Interface
Extract text from publications and split into chunks
Backend Server
Data chunks from user's library of documents
Convert text into vectors and BM25 tokenized library
Context search and Question Relevance Score (QRS) calculations
Context Engineering
User Query
Send LLM response
To calculate and get Answer Relevance Score (ARS) and Hallucination Index (HI)
LLM Server
3 Generation
Answer
Source
Publication title
Page
Response with relevance scores and hallucination index
Retrieved golden context with QRS
Prompt
Q
Context
Payload for LLM
Send query to LLM with chunks as context
LLMs
e.g. Gpt-oss, Llama3.3, Gemma4
Respond with answer, source, relevance scores, and hallucination index
b Context Engineering
Knowledge building
Context retrieval
Data chunks from user's library of documents
User Query
Keyword search
BM25 Algorithm
Tokenize and indexing
Find n chunks with highest 1 BM25 Scores
Tokenized Library
Semantic search
Embedding Model
Store vectors into vector datbase
2 Find n chunks with least Euclidean Distances from query
Vector Database (Chroma*)
Reranker
Top n chunks
Top n chunks
cross encoder
3 Get chunks with hightest Ranking Scores and filter out chunks with low scores
top l chunks based on keywords
top m chunks based on semantic similarity
Merge chunks from two methods
Golden context from best k chunks
QRS calculation using 1 Euclidean Distances, 2 BM25 Scores, and 3 Rerank Scores
QRS
Question Relevance Score (QRS)
Correctness prediction
Golden context from best k chunks
LLM answer
Keyword search
BM25 Algorithm
Tokenize and indexing of k chunks
4 Get BM25 Scores between Answer and chunks
Tokenized context
Semantic search
Embedding Model
Store LLM answer into vector datbase
5 get Eucledian Distances between answer and chunks
Vector Database (Chroma*)
Reranker
k chunks with vector distance and BM25 scores
cross encoder
6 Get Textual Entailment between chunks and the answer { contradiction, entailment, neutral }
Golden context with new scores for answer
ARS and HI calculation using 4 Euclidean Distances, 5 BM25 Scores, and 6 Textual Entailment
ARS
Answer Relevance Score (ARS)
+
HI
Hallucination Index (HI)
*Third Party open-source components

Figure 1. **Overview of the MyGPT workflow**. **a** Top-level view. Blue arrows represent processes followed by user-uploaded documents. Documents are pre-processed, converted from text into chunks, and stored in the backend for “context engineering”. Orange arrows show processes followed by user queries. In the real-time Q&A step, queries are compared against a knowledge base created during pre-processing, appropriate chunks identified as the “golden” context are supplied to the LLM as context for the user query to produce a response (green arrows). The pipeline provides the user with an explicit context source from the library with the generated answer and relevancy scores for question (QRS), answer (ARS), and hallucination index (HI) to identify likely instances of hallucination in the response. **b** In-depth view of MyGPT context engineering. After document chunks are prepared (blue arrow), they are stored in a BM25 (Best Matching 25)-tokenized library for keyword search and in an embedding-based vector library for semantic search. During the Q&A step (orange arrow), the user's query is tokenized to retrieve the top BM25-scored chunks (keyword based retrieval). In parallel, the query is embedded to find the closest vectors (chunk) using Euclidean distance (semantic search based retrieval). The top *n* candidates from each method are selected and are passed to a Reranker (cross-encoder), which assigns a relevance score to each chunk based on both the query and chunk content and filters out chunks with low scores, resulting in *l* chunks from keyword search and *m* chunks from semantic search. The top *k* scored and re-ranked chunks, with duplicates merged, serve as the "golden context", and are supplied to the LLM to generate the response. QRS (Query Relevance Score) is calculated using Euclidean distances from vector search, BM25 scores from keyword search, and reranking scores from the cross-encoder. For correctness prediction (green arrow), the same process is repeated for *k* chunks using BM25 and vector embeddings but compared with LLM response rather than the user’s query. For this step, we used a different Reranker to assess how well the “golden context" supports or contradicts the generated answer, resulting in the Answer Relevance Score (ARS) and Hallucination Index (HI).

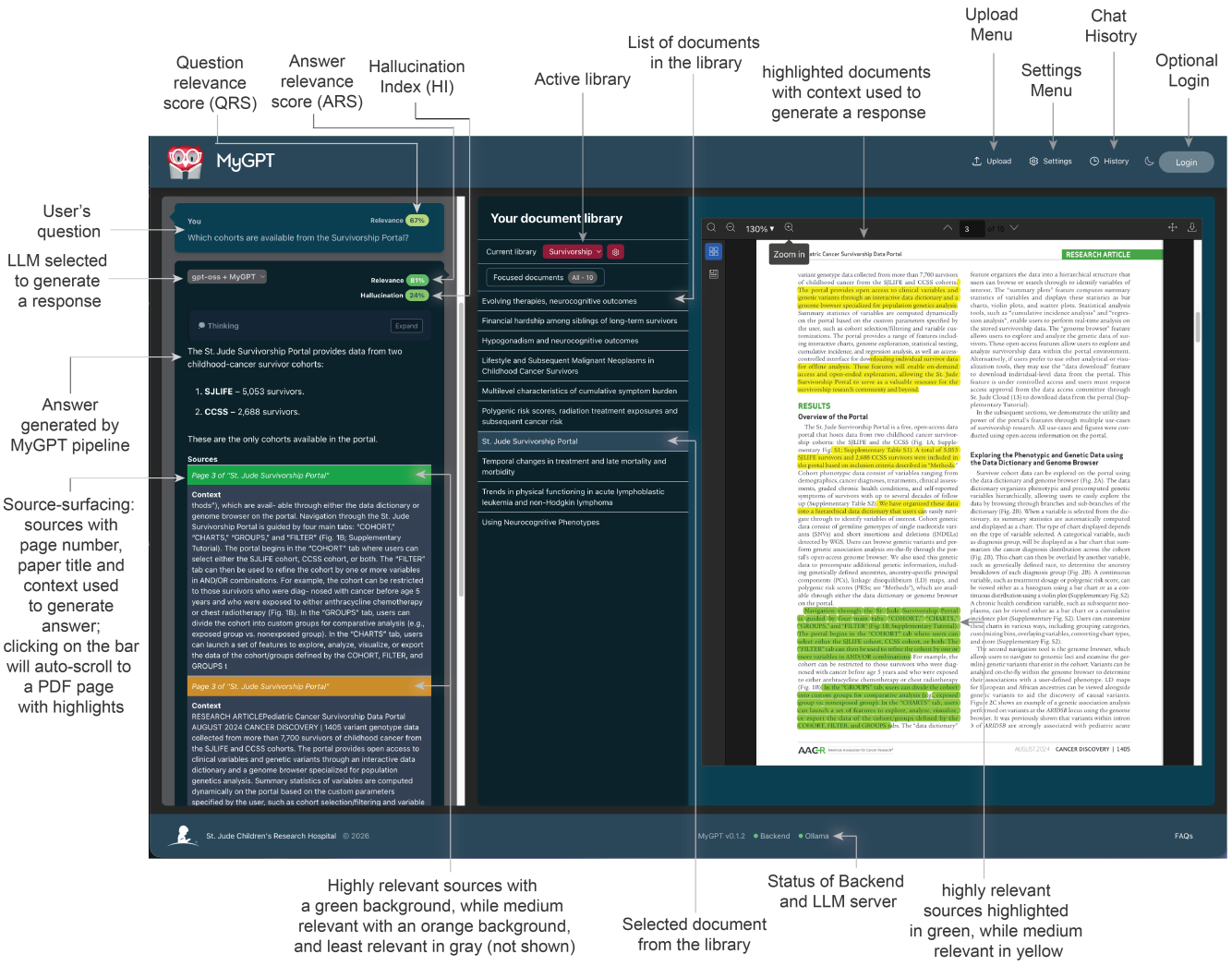


Figure 2. MyGPT user interface. The MyGPT user interface presents three panels. The left-most panel is used for real-time Q&A and displays the user query with a question relevance score (QRS), the LLM generated answer with an answer relevance score (ARS), and a hallucination index (HI). The answer is also accompanied by sources, including the document and specific paragraphs (with exact page numbers) used to generate the answers. Clicking on the source activates the document from the middle panel, which displays all the documents from the library, while the right-most panel displays the document page and the highlighted context. The sources with a green background and green highlights correspond to highly relevant context for the user's questions and the LLM-generated answer, and medium-relevant sources are displayed with an amber background and yellow highlights, helping users to find pages and paragraphs with high significance.

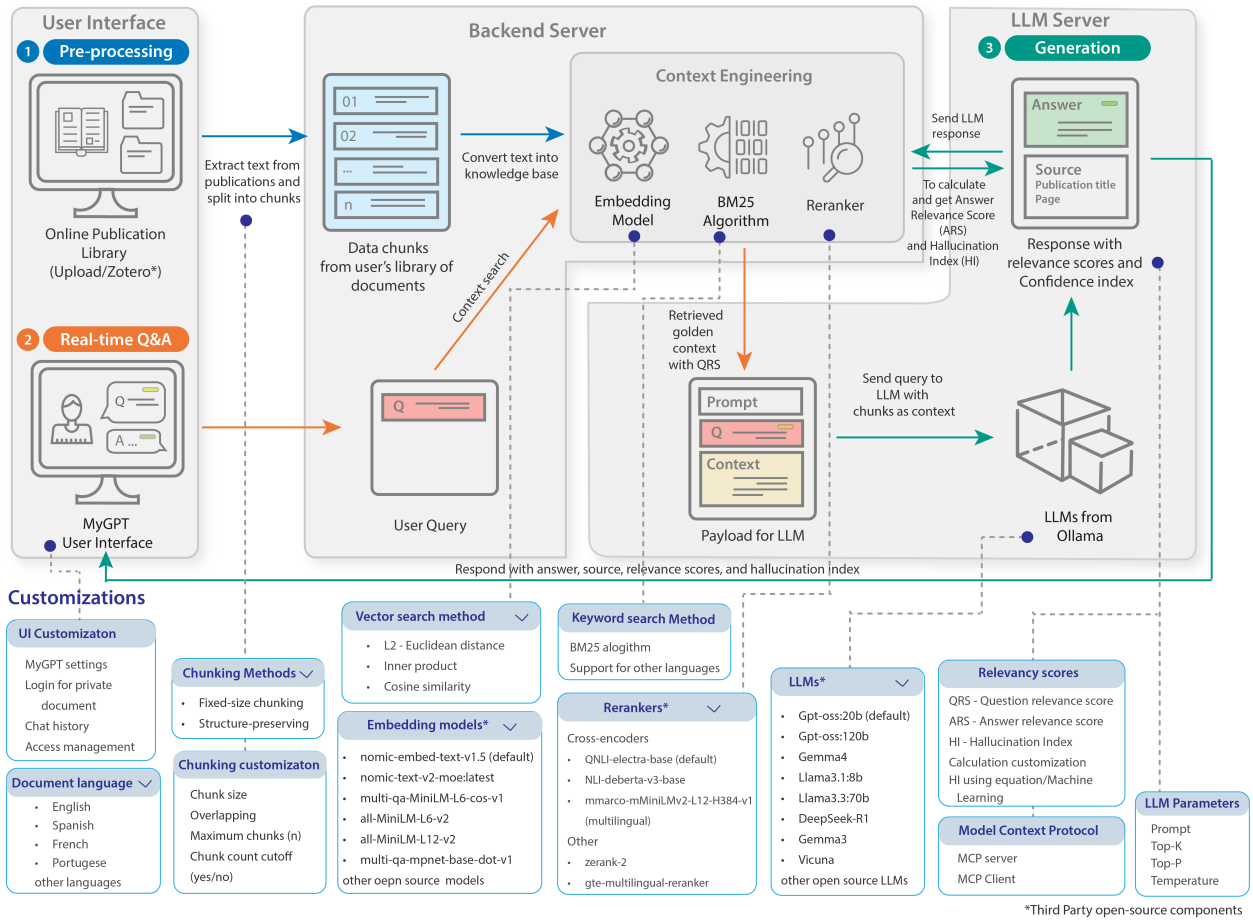


Figure 3. MyGPT pipeline customizations. Users can modify the MyGPT pipeline via the user interface at all three steps of the MyGPT workflow: pre-processing (blue), real-time Q&A (orange), and generation (green). Colored arrows represent the progression of each step. The dark blue dots indicate customization features such as chunking method, embedding models, vector and keyword search methods, Rerankers, relevancy scores, LLMs, creativity parameters and MCP (model context protocol; **Supplementary Information**) capabilities. Dotted lines indicate the customization options at individual steps, with the range of options presented in the connected boxes in the lower panel.

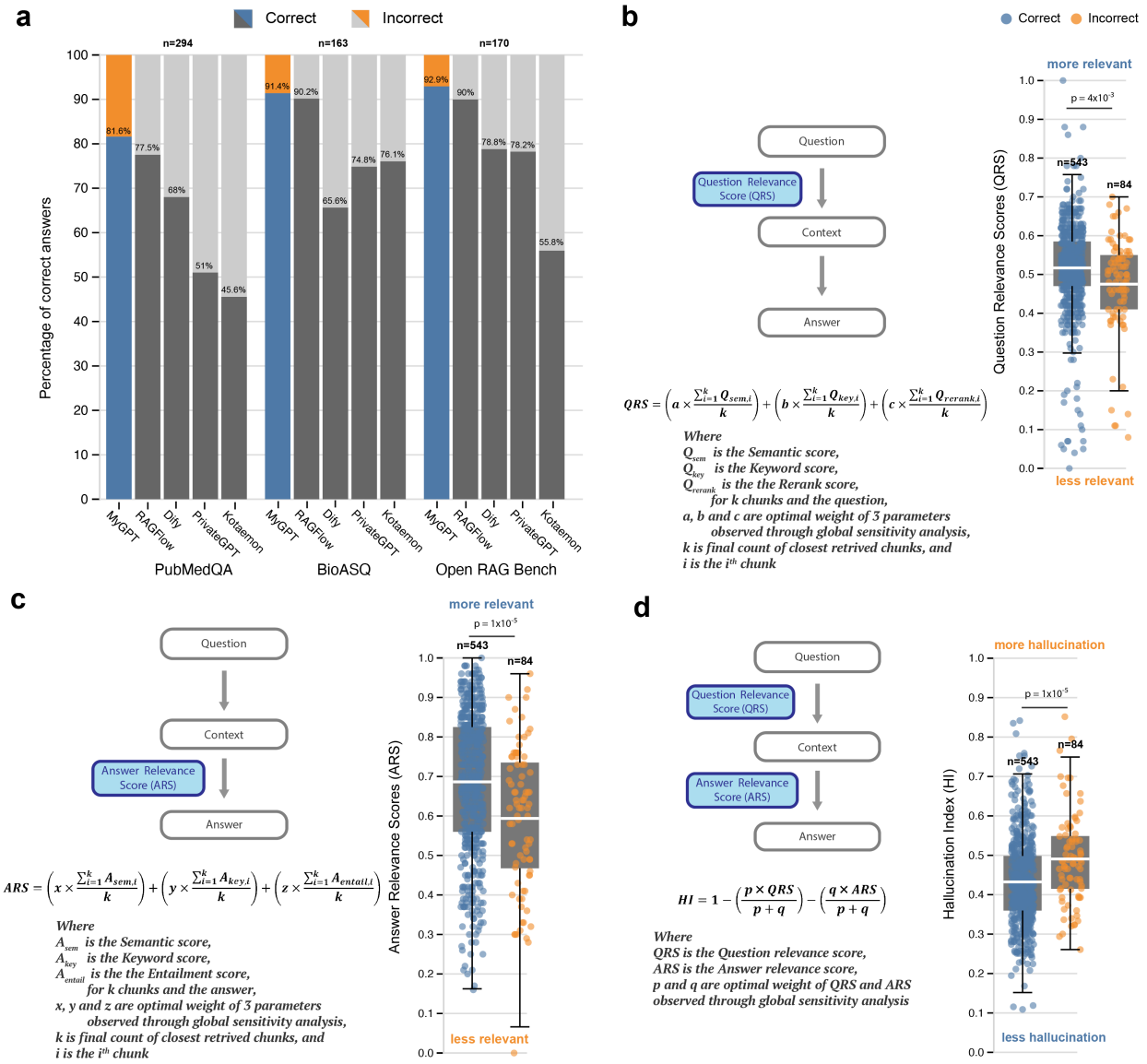


Figure 4. **Benchmarking MyGPT with other open-source RAG solutions using public datasets**. **a** Frequency of answer correctness for MyGPT versus other open-source RAG solutions. A Q&A dataset was collected using three publicly available evaluation datasets: PubMedQA, BioASQ, and Open RAG Benchmark. Using the ground truth provided by the dataset, answers were manually evaluated as "Correct" (dark grey/blue) or "Incorrect" (light grey/orange). A total of 294 questions (294 documents) were collected from PubMedQA, 163 questions (397 documents) from BioASQ, and 170 questions (165 documents) from Open RAG Benchmark. MyGPT performs better than RAGFlow and significantly better than Dify, PrivateGPT and Kotaemon, achieving more than 80% accuracy across all three datasets. **b** MyGPT's Question relevancy score (QRS) distribution for questions of correct and incorrect answers (p = 4x10^{-3}; effect size = 0.11 (biserial rank correlation r, small); Mann Whitney U test) for all three datasets combined shows higher relevance for questions that were answered correctly. **c** MyGPT's Answer relevancy score (ARS) distribution for correct and incorrect answers (p = 1x10^{-5}; effect size = 0.17 (r, small); Mann Whitney U test) combined for all three datasets shows higher scores for the correct than for incorrect answers. **d** MyGPT's default Hallucination Index (HI) for correct and incorrect answers using arithmetic means between QRS and ARS (p = 1x10^{-5}; effect size = 0.17 (r, small); Mann Whitney U test) combined for all three datasets shows significantly higher value for the hallucination index for incorrect than for correct answers.

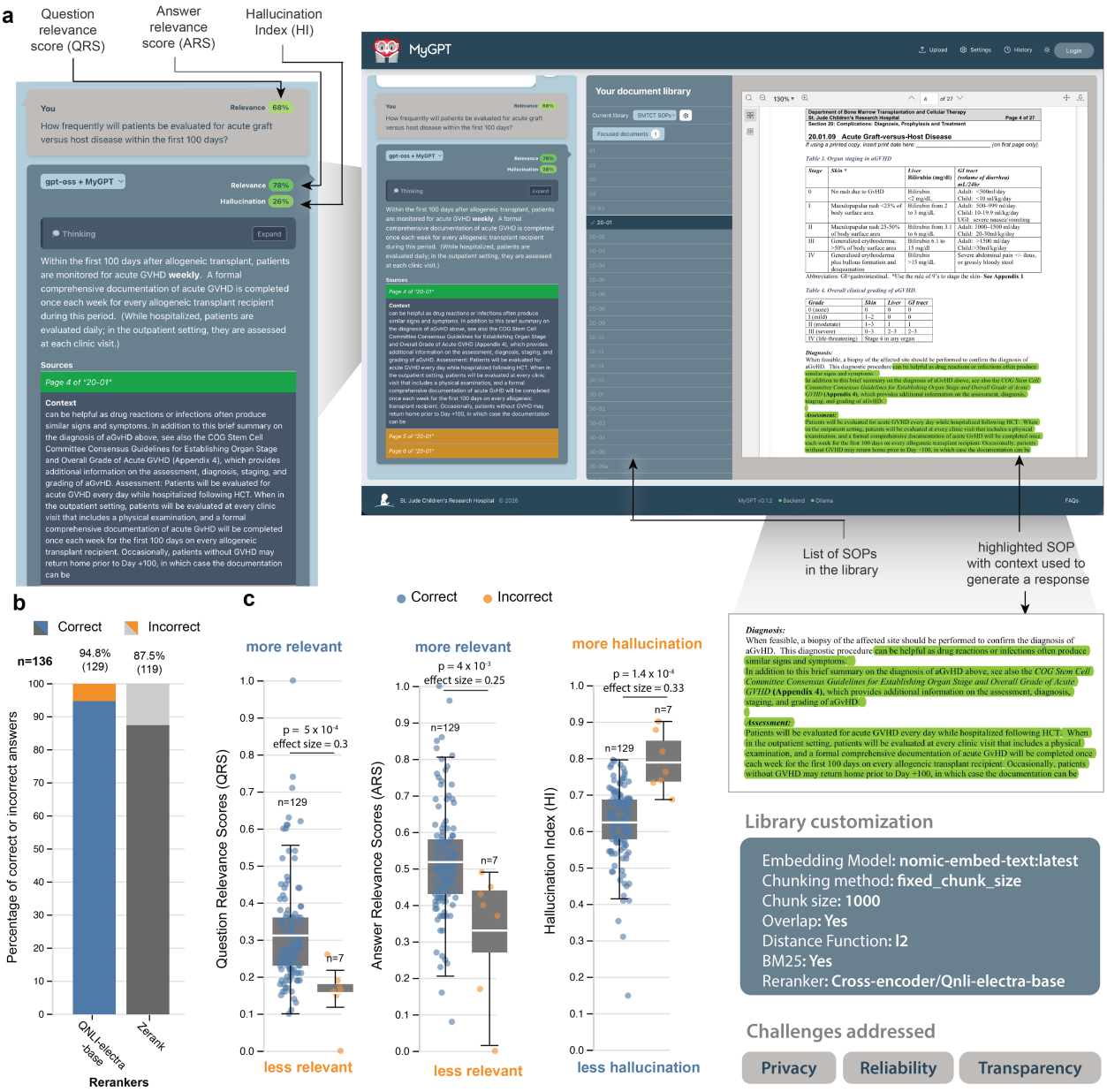


Figure 5. **Use case: Standard Operating Procedures (SOPs) for Bone Marrow Transplantation. a** Example of a question and answer using the library of private SOP documents from the Bone Marrow Transplantation Library at St. Jude. The example shows the user's question and answer, along with the highlighted section of the SOP used by LLM (gpt-oss:20b) to generate the answer. The library was created using the "nomic-embed-text" embedding model, which incorporates BM25 for precise keyword search, in addition to context-based semantic search via a vector database and reranked using "cross-encoder/qnli-electra-base" Reranker to filter out less relevant chunks. **b** Of the 136 questions (with ground truth) collected from 68 documents for this evaluation, MyGPT was able to answer 129 questions correctly with 94.85% accuracy. Answers were also collected using Zerank Reranker, which was able to answer 119 answers correctly with 87.5% accuracy, showing that "qnli-electra-base" is significantly better (p = 0.01; effect size = 0.56 (phi coefficient Φ, large); McNemar chi-square test). **c** MyGPT's Question Relevance Score (QRS) distribution for questions with correct or incorrect answers (p = 5x10^{-4}; effect size = 0.3 (r, small); Mann

Whitney U test) shows higher relevance for questions that were answered correctly. Similarly, MyGPT's Answer Relevancy Score (ARS) distribution for correct and incorrect answers ($p = 4x10^{-3}$; effect size = 0.25 (r, small); Mann Whitney U test) shows higher scores for the correct than incorrect answers. MyGPT's Default Hallucination Index (HI) for correct vs incorrect answers ($p = 1.4x10^{-4}$; effect size = 0.33 (r, medium); Mann Whitney U test) shows significantly higher hallucination for incorrect answers.

a

	Users	Documents	Use cases	Question types
	Researchers	Publications Research talks Electronic notebooks	Literature review Talk summary Experiment summary	Summarization Keyword Yes/no Data query Complex question
	Departments	Collaboration material Reports	Project on-boarding Reduce domain-specific barriers	Summarization Complex question
	Hospitals	Electronic Health Records (EHR) Patient care records	Summarize health records Patient on-boarding Personal assistant	Summarization Yes/no Data query
	Institutes	Internal wiki Project and annual reports	Internal information-seeking Orientation Quicker key insights	Summarization Complex question Data query
	Human Resources and Administration	Work Policies Job Posts Candidate Resumes	FAQs Institutional chat-bot On-boarding assistant Talent acquisition	Summarization Yes/no Data query
	Students	Lectures Reference books Digital textbooks Educational videos	Deeper understanding of subjects Reviewing topics Different materials and formats	Summarization Complex question Keywords
	Commercial settings	Confidential material Legal documents	Secure and private Q&A Faster information retrieval Faster key Insights	Yes/no Data query
	General public	Personal journals and notebooks	Personal knowledge Exploration and idea generation	Summarization Complex question

b

Solution	Business model	End-to-end solution	Source highlighting	Confidence metrics	Multilingual support	Open-source Reranker support	Commercial GPTs Support
MyGPT	Open source	✓	✓	✓	✓	✓	✗
RAGFlow	Open source	✓	✗	✗	✓	✗ (Commercial Reranker)	✓
Dify	Open source	✓	✗	Limited (question relevance)	✗	✗ (Commercial Reranker)	✓
PrivateGPT	Open source	✓	✗	✗	✗	✓	✓
Kotaemon	Open source	✓	Limited	Limited (per chunk)	✗	✗ (Commercial Reranker)	✓
Verba	Open source	✗ (Front-end only)	✗	✗	✗	✗	✓
LlamaIndex	Semi Open source	✗ (Python package)	N/A	✗	✓	✗	✓
BioChatter	Open source	✓	✗	✗	✗	✗	✗
ChatPDF	Commercial	✓	✓	✗	✓	✗	✗ (OpenAI only)
NotebookLM	Commercial	✓	✗	✗	✓	✗	✗ (Gemini only)

Figure 6. MyGPT applications and comparison to current GPT solutions. **a** Example use cases of MyGPT for information-seeking tasks performed by individual users or groups of users from academia, industry, and the general public. Users have the flexibility to cusromize and modify MyGPT parameters based on their specific use case. **b** Comparative summary table of MyGTP's features with those of various commercial and open-source RAG solutions. MyGPT provides a software solution that fills the unmet needs of existing GPT versions in a single platform.

METHODS

MyGPT architecture

The architecture of the MyGPT workflow comprises three sections: (i) a user interface (UI), (ii) a backend server, and (iii) an LLM server (**Fig. 1**). The UI and backend serve a dual purpose of library pre-processing and real-time question and answer.

For library pre-processing, the UI receives the documents and customizations for the chunking method, the selected embedding model, and the distance function and sends them to the backend. The backend then converts documents into chunks and the chunks into embeddings using a user-selected embedding model and stores them in a vector database for semantic search. For different libraries, the backend creates separate sections/collections and stores them in the same vector database along with necessary metadata about the document type, title of documents, and user. This enables multiple users to use the same library and vector database without the need for access control and prevents leakage of information among libraries of individual or multiple users. In addition to semantic search, MyGPT also creates BM25 tokenizer for each library for keyword search. The UI also provides an interface to manage libraries: users can add, switch, or delete the documents in the libraries. Deleting the library deletes all the documents, vector embeddings and tokenizer from the vector database and removes the library chat history from the backend database.

For real-time question-and-answer, the UI receives the question and sends it to the backend with a default prompt. The backend server converts the question into an embedding (using the same embedding model as the one used for processing the library) and sends the encoded question to the vector database to retrieve the nearest chunks (see below). MyGPT also collects chunks identified by the BM25 algorithm that match the keywords in the question and then combines chunks identified by both methods by merging duplicates, filtering and re-ranking them using Reranker (see below). The final set of chunks is provided to the LLM server as context for a response. We have incorporated several open-access LLMs into the MyGPT pipeline, including Gpt-oss, Llama3, Llama2, Mistral, Vicuna, and Gemma, with Gpt-oss set as the default option (**Supplementary Table 1**). The main purpose of these LLMs is to receive a prompt, a user's query, and the context and to respond with an LLM-generated answer with source citing from the documents in the library (**Fig. 1**). The UI also provides options to change the default prompt or the creativity vs precision settings before the LLM generates the answer. The LLM section of the pipeline can run independently of MyGPT and be used for purposes beyond MyGPT. In addition, a single LLM instance can power multiple instances of MyGPT to reduce cost and maintenance.

The MyGPT pipeline uses the following third-party, open-access tools (**Fig. 3**) to orchestrate user interaction with LLMs for the three sections mentioned above:

- **Reference management tool (UI)**: We provide the option to create the document library using Zotero[12], a free reference management tool for collecting and annotating publications. Zotero users can create a private or shared library of publications. Alternatively, MyGPT users can upload documents directly without using any reference management tool.
- **Embedding model (Backend)**: We utilize "sentence transformer", a Python library that converts text, paragraphs, or images into vectors and Ollama's capability to host and inference embedding models. In our analysis, nomic-embedding-text hosted using Ollama showed ideal performance for different datasets, and we use it as a default. MyGPT also provides an option to use the multi-qa-MiniLM-L6-cos-v1[6] sentence transformer, an open-access embedding model available in the Chroma database. MyGPT can also be used with other embedding models available via Ollama and Hugging Face.
- **Vector database (Backend)**: We use Chroma database[44], an AI-native open-source vector embedding database, to store all the chunks from user-provided documents locally as vectors, allowing for easy and fast searching within the vector database.

- **BM25 (Backend)**: We use a lightweight Python library, BM25S, to create a keyword search tokenizer for the uploaded libraries. We use PyStemmer's stemming functionality with BM25S to support English (default) and other languages such as Spanish, French, and Portuguese. If the language of the document is selected as "unknown" when uploading the library, MyGPT will skip the BM25 step.
- **Reranker (Backend)**: We used the Cross-encoder library from "sentence transformer" for reranking. We used two different Rerankers: one for reranking chunks based on the user's question and another for computing textual entailment between the retrieved chunks and the LLM-generated answer. To compare questions with consolidated chunks from semantic and keyword searches, we used "cross-encoder/qnli-electra-base" trained on Wikipedia passages and questions that could be answered by using those passages. To get textual entailment of those same chunks with the LLM-generated answer, we used "cross-encoder/nli-deberta-v3-base" as the default Reranker. We also tried several other Rerankers available via Huggingface, supported by the Cross-encoder library, and available via UI. The Cross-encoder library allows downloading and use of a local Reranker, enabling both faster performance than querying the Hugging Face-hosted Reranker for each real-time Q&A as well as offline usage of Reranker after initial download.
- **LLM hosting library (LLM server)**: We use Ollama[23], an LLM hosting library that enables users to employ open-source LLMs locally without the need to connect to proprietary online services. MyGPT employs an API architecture to utilize open-source LLMs that are currently available via Ollama or will be made available in the future. As of this publication, we recommend Gpt-oss:20b for personal laptops and Gpt-oss:120b for Ollama installed on servers/VMs. MyGPT also provides a user interface (UI) to install other LLMs such as Llama2, Gemma, Gemma2, Vicuna, and Mistral. Once installed, the MyGPT UI will display the various LLMs available for use in the pipeline (**Supplementary Fig. 3**).

Document chunking methods

MyGPT provides two methods for chunking text:

- **Fixed-size chunking**: MyGPT divides the text from the user-provided documents into chunks of a predetermined number of characters from start to end. By default, MyGPT uses a chunk size of 1000 characters, but users can choose from options of 500, 750, 1000, or 1200 characters. Additionally, users can adjust the chunk overlap, which refers to the number of characters shared between two consecutive chunks. If this option is selected, MyGPT automatically applies a 20% overlap between adjacent chunks to maintain the meaning of sentences at the end of each chunk.

  Document-structure-preserving recursive chunking: MyGPT first identifies major hierarchical structures within the documents, including sections, subsections, and other semantic elements. This is achieved by leveraging the table of contents available in the PDF's metadata, or, in the absence of this information, by utilizing the GROBID[45] document parsing library. GROBID's XML outputs provide structured boundaries that preserve the organization of academic publications. Once the sections are identified, each section is treated as an individual entity for subsequent chunking. Unlike fixed-size chunking, this approach first identifies paragraphs within each section. The recursive approach attempts to split the text at subsection boundaries, then paragraph breaks, and finally sentence boundaries. If a paragraph exceeds the chosen chunk size (e.g., 1000 characters), it will be further split without breaking sentences mid-way. This method also allows users to configure overlap for adjacent chunks. By considering the document hierarchy, this approach maintains semantic coherence at the most significant structural levels while adhering to the defined token limits.

Vector embedding of textual chunks and identification of chunks for semantic search

MyGPT converts chunks of text from the library into high-dimensional vectors. MyGPT splits the text from the provided documents with one of the methods mentioned in the prior section, then converts these chunks into vectors using multi-qa-MiniLM-L6-cos-v1 or a user-selected embedding model. The vector dimensions are different for different embedding models; for example, it is 384 for multi-qa-MiniLM-L6-cos-v1 and 768

for nomic-ai/nomic-embed-text-v1. When a question is asked, MyGPT converts the question into a vector using the same embedding model, embeds it in the same vector space as the library, and then finds the nearest *n* chunks (**Fig. 1**). The number of chunks, *n*, is set to 5 by default, and users can change the value of *n* in the user interface settings. If MyGPT can't find all *n* chunks, or if not all nearest chunks meet the cutoff criteria of vector distances, chunks returned from semantic search will be less than *n*, where *n* is the maximum number of chunks from semantic search. MyGPT converts vector distances to Semantic score Q_{sem} via min-max normalization using the observed minimum and maximum distances for public datasets (see main text). Once the LLM generates the answer, MyGPT converts the answer into a vector using the same embedding model, finds the distance between the answer and *k* chunks from the "golden context", and calculates Semantic score A_{sem} via min-max normalization as above for Q_{sem}. Note that the *k* chunks for the "golden context" are finalized during *Reranking and the consolidation of chunks* step, as discussed in the following sections, and, once finalized, the "golden context" with *k* chunks is used for the remaining steps in the workflow.

Keyword search for textual chunks using BM25

MyGPT offers the option to use the BM25 method to identify relevant chunks through keyword search, ensuring that no keywords are overlooked. If selected alongside vector-based contextual search, MyGPT will collect *n* chunks with the highest BM25 scores for the user query. Similar to semantic search, if MyGPT can't find *n* chunks or not all *n* chunks meet the BM25 cutoff score, returning chunks from keyword search will be fewer than *n*, where *n* is the maximum number of chunks returning from keyword search, making the maximum number of chunks returning from both methods *2n* (**Fig. 1**). MyGPT converts BM25 raw scores to Keyword score Q_{key} via min-max normalization using the observed minimum and maximum BM25 scores for public datasets (see main text). Once the LLM generates the answer, MyGPT calculates the BM25 scores between the answer and *k* chunks from the "golden context" and generates Keyword score A_{key} via min-max normalization as above for the Q_{key}.

Reranking and consolidation of chunks for LLM context

Once the semantic and keyword search returns chunks, those chunks are provided to Reranker along with the user's query. The Reranker assigns a rerank score, Q_{rerank}, and a rank based on the probability of answering the user's query using that chunk. Chunks that do not meet the minimum cutoff for Rerank scores will be removed, leaving *l* chunks from keyword search and *m* chunks from semantic search. These chunks from both of these methods are consolidated by merging duplicates, leaving *k* chunks to be returned as the "golden context" for the rest of the MyGPT workflow (**Fig. 1**).

Sample size calculations using Power analysis

Because the RAG systems are evaluated on the same questions, and multiple questions may come from the same document, standard independent-sample power analysis is not appropriate. We therefore perform a sample size calculation (i.e., number of documents) that accounts for both paired evaluations and clustering within documents.To determine the ideal sample size for performance comparisons between different RAG-applications or RAG pipeline settings, we used the following formula,

$$c = \frac{\left(z_{1-\frac{\alpha}{2}} + z_{1-\beta}\right)^2 \left\{p_1(1-p_1) + p_2(1-p_2) - 2\rho\sqrt{p_1(1-p_1)\,p_2(1-p_2)}\right\}\{1+(m-1)\rho_c\}}{m\,(p_1-p_2)^2} \quad (8)$$

Where

z is the z-score

α is Type I error rate; e.g., α = 0.05 and $z_{1-\frac{\alpha}{2}}$ = 1.96,

β is Type II error rate = 1 – Power, e.g., β = 0.1 and $z_{1-\beta}$ = 1.28,

p_1 and p_2 are model accuracies for models 1 and 2; e.g., p_1 = 0.6 and p_2 = 0.4,

m is number of questions per document, e.g., m = 5,

c is number of documents,
Y_{1ij} and Y_{2ij} are binary correctness indicators for models 1 and 2, respectively, on question j from document i, where 1= correct and 0= incorrect,
ρ is correlation between models on the same question (i.e., how similarly the models succeed or fail on a question), defined as $Corr(Y_{1ij}, Y_{2ij})$, where i is document index and j is question index; e.g., ρ = 0.8,
ρ_c is the intra-document correlation of the paired differences across questions within the same document, where $D_{ij} = Y_{1ij} - Y_{2ij}$, and $\rho_c = Corr(D_{ij}, D_{ij'})$; e.g., ρ_c = 0.5,

This formula contains two correlation adjustments: one for pairing across models (ρ) and one for clustering within documents (ρ_c).

The pairing correction term $2\rho\sqrt{p_1(1-p_1)\,p_2\,(1-p_2)}$ accounts for pairing between the two models when evaluated on the same question. When ρ =0, there is no pairing effect, and the setup behaves like two independent sets of questions assigned to two independent models. Positive pairing correlation reduces the variance of the paired difference and therefore decreases the required sample size.

The clustering correction term $\{1 + (m-1)\rho_c\}$ accounts for clustering within documents, where multiple questions are nested within the same document. When ρ_c = 0, questions within documents are independent, indicating no clustering and reducing to an independent-sample design. When m=1, there is only one question per document, so clustering does not occur. Positive clustering inflates the variance of the estimated mean difference and therefore increases the required sample size.

In the absence of both clustering and pairing, the formula reduces to the standard sample size calculation for comparing two proportions under an independent-sample design.

While p_1, p_2, ρ, ρ_c and m may vary across evaluation datasets, use-cases, MyGPT settings and comparisons with other RAG solutions, we fix the Type I and Type II error rates at 0.05 and 0.1, respectively, corresponding to a 5% significance level and 90% power. These choices provide a conservative basis for determining the number of documents required across datasets. The parameter values for individual datasets are specified in their corresponding sections (*Benchmarking MyGPT and other open-source RAG solutions using public datasets, MyGPT libraries for the different use cases,* and **Supplementary Table 5**).

Running and hosting MyGPT
The MyGPT pipeline can be run on a local computer for personal use or hosted on a server/cloud infrastructure for group, department, or institution-wide usage.

To run MyGPT locally, a user needs a device with at least 8 CPUs, 16 GB RAM, and a GPU. We have provided benchmarking instructions for testing Ollama on personal devices (see GitHub repository for instructions). If users can run Ollama satisfactorily, they can install MyGPT and run the entire pipeline on the same device. Detailed descriptions of the steps for all approaches are available in the GitHub repository (https://github.com/mb-group/MyGPT_public). We have tested installation and LLM performance of MyGPT on various personal devices (**Supplementary Table 2**). During our testing, we were also able to install and run MyGPT on an affordable Edge AI compute device – NVIDIA Jetson Orin Nano (cost - $250), with small LLMs such as Llama 3.2 (1b, 3b) and Gemma (2b) (**Supplementary Fig. 7**).

In cases with many MyGPT users, such as departments, institutions, or industry settings, hosting MyGPT on a server or cloud platform may be advantageous. To optimally run the MyGPT pipeline in such environments, the installer needs to understand the operation and maintenance of a web application server. We have tested MyGPT on local servers with various GPUs, including Nvidia GPU GeForce RTX 4090 and Nvidia GPU A100, as well as on cloud infrastructures such as Amazon Web Services. As resource

requirements and configurations will vary for each use case (details in **Supplementary Table 1**), involving Information Technology personnel to assist in hosting MyGPT for these instances may be helpful.

Cost and resource usage

To illustrate the range of deployments and resource considerations, we hosted MyGPT on personal devices, on commodity hardware for group use, in the cloud for review purposes, and on a VM for institutional use.

- **Personal devices**: Installation on Mac laptops with M1 to M4 chips ranged from $2,000 - $4,500 and were suitable for experimental scientists and AI engineers. Users followed GitHub instructions and maintained their installations by periodically updating code from GitHub.
- **Commodity hardware**: MyGPT was hosted for 40 team members on Nvidia DGX-1s at $4,000. Installation by a senior engineer took 2 hours. Recommended similar hardware and LLM hosting configurations are listed in **Supplementary Table 6**.
- **On Cloud:** MyGPT was hosted on Azure, with an LLM server VM (NC8as T4 v3 - 8 vCPUs, 56 GB RAM) at $548.96/month and backend/frontend VM (E4ads v5 - 4 vCPUs, 32 GB RAM) at $325.58/month. This and similar estimates and configurations are in **Supplementary Table 3**. Installation by three team members took 12 hours, and maintenance requires 20-30 minutes every 3 months.
- **Institutional deployment:** MyGPT was deployed at St. Jude Children's Research Hospital on on-premises VMs approved for Protected health information (PHI) data storage. The process began with obtaining necessary approvals from institutional stakeholders. Our team of 4 professionals (2 Sr. Software engineers, a Software Architect, and an AI engineer) performed installation, testing, and vulnerability and security checks. The Information Security and infrastructure teams reviewed compliance. Deployment spanned 4 weeks for these steps. Maintenance involves updating code and LLMs/embedding models every 2-3 months, requiring 0.5 - 1 hour. We use 4 A100 GPUs for the LLM server and a VM with 4 CPUs, 64 GB of RAM, and 1 TB of storage for the backend and frontend. Due to confidentiality and vendor contracting, VM costs cannot be shared publicly.

Cost comparisons with other commercial and open-source RAG solutions are provided in **Supplementary Table 7.**

Finding the optimal setting of QRS, ARS, and HI

For public datasets and individual use case, we calculated optimal weights for QRS, ARS, and HI equations using global sensitivity analysis. Global sensitivity analysis explores how outputs vary across the entire range of possible input values.

- QRS is calculated by multiplying optimal weights a, b, and c by Q_{sem} (Semantic score), Q_{key} (Keyword score), and Q_{rerank} (Rerank score), respectively, and summing the three values, as per the equation (1). To obtain the optimal weights a, b and c, we assigned 10 different weights (-4 to 5) to Q_{sem}, Q_{key} and Q_{rerank} and calculated the area under Precision-Recall Curve (PR-AUC) for each combination of these three variables. We plotted them in ten (10x10) heatmaps, each for individual rerank weights, x-axis with 10 vector weights, and y-axis with 10 BM25 weights (**Supplementary Fig. 8**). We selected optimal weights for all 3 variables with the highest PR-AUC value. For 3 public datasets combined (n=627), the optimal weights a, b, and c are 5, -2, and -2, respectively, with PR-AUC of 90%.
- ARS is calculated by multiplying optimal weights x, y, and z by A_{sem} (Semantic score), A_{key} (Keyword score), and A_{rerank} (Rerank score), respectively, and adding the three values per equation (4). We calculated optimal weights of 3 variables using the same method as for QRS by plotting 10 different weights in ten 10x10 heatmaps (**Supplementary Fig. 8**). For 3 public datasets combined, the optimal weights x, y and z are 5, -2, and 0, respectively, with PR-AUC of 92%.
- HI is calculated by multiplying optimal weights p and q by QRS and ARS, respectively, per equation (7). We calculated them using the same method, giving optimal weights of 2 for p and 3 for q to for the public datasets combined.

Benchmarking MyGPT and other open-source RAG solutions using public datasets

We used 3 different public datasets to benchmark MyGPT and other RAG solutions:

- **PubMedQA:** The dataset contains 1.0k Q&A pairs, with answers labeled as yes, no, or maybe, generated by humans and suitable for automated evaluation. Each of the questions can be answered from a single published article. With institutional access to published articles, we collected 294 articles in PDF format. The number of documents (294) was ideal for comparison with other RAG solutions, such as Dify, PrivateGPT, and Kotaemon, based on sample-size calculations using power analysis (**Supplementary Table 5**). Given the close correctness rates for MyGPT and RAGFlow, the required number of documents (806) is higher than the number we used, and we are unable use more documents due to limitations on the number of published articles we can obtain with institutional access.
- **BioASQ:** BioASQ provides human-curated pairs of Q&A with exact answers and necessary context as snippets from the original published article. We used Training dataset 13b to collect 163 questions and answers. We created 210 PDFs with abstracts/snippets provided by the datasets and used them as a library of documents. The questions were independent, and no clustering was present as a BioASQ dataset design criterion. Similar to PubMedQA, the number of documents (210) was ideal for comparison with other RAG solutions, such as Dify, PrivateGPT, and Kotaemon, but for RAGFlow we will need 5106 documents based on sample-size calculations. It is possible to create additional PDFs using the same method we used for the 210 PDF documents and perform the evaluation, but efforts for manual evaluation are beyond the scope of this manuscript.
- **Open RAG benchmark:** This dataset provides Q&A pairs from 1.0k Arxiv documents, with 400 positive and 600 negative documents. As a subset, we collected 170 questions and 165 PDFs corresponding to the ArXiv documents referenced. This dataset provides 2 types of text-only query questions: abstractive and extractive. Abstractive queries require generating a summary or a rephrased response using understanding and synthesis, while extractive queries require fact-based extraction of an answer directly from the given text. Of 170 Q&A pairs, we used 78 abstractive and 92 extractive queries. Similar to the two prior datasets, the number of documents (170) was ideal for comparison with other RAG solutions such as Dify, PrivateGPT, and Kotaemon, but for RAGFlow, we would need 1063 documents based on sample-size calculations. Per the dataset design, we can increase the document count up to 400 with 1900 Q&A, and the time and effort of manual evaluation is beyond the scope of this manuscript.

Benchmarking and usability testing of the MyGPT pipeline

To test the MyGPT pipeline for technical and usability aspects, we asked 10 participants with varying domain knowledge and technical skills to install and test MyGPT on their personal computers and to use it with their respective libraries. We further assessed the performance of MyGPT by running it in various environments such as on a personal computer, a local server with GPU inferencing, and cloud infrastructure. Differences in response times are reported in **Supplementary Table 2**. During our benchmarking test, we noted that the Gemma model was the fastest in all environments, but the answers were generally shorter and provided fewer details than those generated by larger models such as Llama3:70b. Comparing different environments, the performance of MyGPT on an Apple laptop with M1 GPU (8 CPUs and 64 GB RAM) was equivalent to or better than Google Cloud Platform (GCP) with a T4, L4 or A100 GPU and cloud infrastructure such as Amazon Web Services (AWS) with A10G GPU. Users without access to GPU via a PC or laptop or who have less powerful laptops can use either compact Edge AI devices such as Nvidia Jetson Orin (starts at $250) or use Cloud service with T4 GPU, which can cost around $0.42 per GPU per hour or around $306 per GPU per month on-demand (in US market for AWS as of April 2026).

Library size benchmarking

We tested different library sizes and their impact on the MyGPT workflow. Rather than focusing on the number of documents, we examined pages per library, creating four libraries with 10, 100, 1,000, and 10,000 pages. These were tested on two Apple MacBooks with M1 and M4 GPUs (8 CPUs, 64 GB RAM) and a St. Jude Server with A100 GPU (64 CPUs, 2 TB RAM). Uploading and retrieval times were managed

by the MyGPT backend, while generation time was handled by the LLM server. The end-to-end timing captured the complete Q&A process. Uploading was performed using a fixed-chunking method (1,000-character chunks) with the nomic-embed-text-v1.5 model for embedding. We used llama3:8b for generation on both laptops and the server. Results are reported in **Supplementary Table 4**.

MyGPT libraries for the different use cases

To demonstrate the application of MyGPT for biomedical research, clinical tasks and global health policy at St. Jude Children's Research Hospital, we created multiple instances of MyGPT.

- For the literature review of biomedical research articles, we created a library of articles on various kinases. The library contained 83 biomedical publications for 26 different kinases (99 kinase-publication pairs), and we created 297 Q&A pairs with input from an expert in the field (provided as **Supplementary material**). The library was created using the nomic-embed-text:latest embedding model with a fixed chunking method of 1000 characters and an overlap of 20%. The Squared L2 distance function and the BM25 algorithm were used for keyword matching, and retrieved chunks were reranked using "cross-encoder/qnli-electra-base" Reranker. We utilized gpt-oss:20b LLM to generate an answer. For the first experiment, a default chunk count of 5 was used as the maximum number of chunks returned by the semantic and keyword search. For the second experiment to test the hypothesis that complex questions benefit from a larger number of chunks, we used n=15, which yielded more chunks for most questions and ultimately more correct answers (**Supplementary Fig. 4**).
- For the clinical document, we created a MyGPT library with Standard Operating Procedures (SOPs) from the Department of Bone Marrow Transplantation and Cellular Therapy (BMTCT) at St. Jude. The library contained 136 questions (with ground truth) from 68 SOP documents and was created using the nomic-embed-text:latest embedding model with a fixed chunking method of 1000 characters, an overlap of 20% between chunks, a Squared L2 distance function, and the BM25 algorithm for keyword matching. Retrieved chunks were reranked using "cross-encoder/qnli-electra-base" Reranker, and gpt-oss:20b LLM was used to generate an answer.
- For Q&A on multilingual global health policy documents, we created Spanish, French, and English document libraries, each containing 10 health policies from the St. Jude Global Pediatric Medicine Department. Using nomic-text-v2-moe:latest, we embedded texts with a fixed chunk size of 1000 characters and 20% overlap, using the Squared L2 distance and the BM25 algorithm for each language. Due to limited foreign language expertise and manual validation, we omitted the Reranker from this evaluation. For answers, we used gpt-oss:20b. To support multiple languages, we modified MyGPT in several ways: during document upload, users must select the language of documents, prompting MyGPT to choose the appropriate embedding model and BM25 version. User questions are translated into the document's language using TranslateGemma LLM before semantic and keyword search for context retrieval. Using this context, the user-selected LLM generates a response in the user's language, and MyGPT uses translated answers from TranslateGemma to compute ARS and HI. Current testing confirms this modified pipeline works as intended, though future evaluation of Reranker suitability is needed.

Data sources and preprocessing for a machine learning model to compute Hallucination Index

We built a random forest classifier[46] to predict the incorrectness of each question-answer pair based on various associated scores, such as semantic, keyword, and reranking scores. The training dataset consists of 3 public and 3 private datasets (see the above section and the main text), totaling 1045 question-answer pairs. Each question-answer pair has semantic, keyword, and reranking scores for the question and the answer, as well as a binary correctness label. The correctness labels for public datasets were provided by two independent researchers based on ground-truth answers available with the datasets, and the labels for private datasets were provided by experts in the domain for real-world use cases.

Semantic scores, BM25 scores, and rerank scores were provided as a list for each Q&A, with each element containing the score for the corresponding chunk. For each list, we computed the summary statistics mean, maximum, minimum, and standard deviation. These statistics were computed for question semantic scores, question BM25 scores, question rerank scores, answer semantic scores, and answer BM25 scores, totalling 20 features for the training data. Q&A were dropped if any of the following fields were missing: semantic scores for questions, BM25 scores for questions, reranked question scores, semantic scores for answers, BM25 scores for answers, or reranking sentiment labels for answers. The binary outcome variable (y_true) was defined as incorrect = 1 and correct = 0, derived from the correctness label.

Architecture of a random forest model and model Interpretability with SHAP

Using the training data mentioned in the above section, we trained a random forest classifier to predict incorrectness (binary classification). As we observed significantly higher HI for incorrect answers using the HI equation, we used incorrectness as the primary prediction target rather than correctness. The model was implemented using scikit-learn[47]. To assess robustness to data splitting, we performed 10 independent replications using different random seeds. In each replication, the dataset was split into training (80%) and test (20%) sets using stratified sampling. A Random Forest model was then trained on the training set and evaluated on the test set. In addition to the default probability cutoff of 0.50, we explored optimized decision thresholds. For each test set, predicted probabilities were thresholded across values from 0.01 to 0.99. For each threshold, an ROC-AUC was computed using thresholded predictions. The threshold yielding the highest valid test-set ROC-AUC across replications was recorded as the best cutoff.

The best-performing model across replications was selected based on test ROC-AUC. Feature importance analyses were performed using the built-in feature importance function of Random Forest in scikit-learn, and SHAP[48] (SHapley Additive exPlanations) feature importance values using the selected model. To assess feature stability, we computed feature importances for each replication and evaluated pairwise correlations between importance vectors across replications. Model interpretability was evaluated using SHAP with a TreeExplainer optimized for ensemble tree models. SHAP values were computed for all samples, focusing on class 1 (incorrect prediction) contributions.

Acknowledgments
We thank ALSAC for funding and support. We also acknowledge support from the Blue-Sky Kinase initiative (J.P., J.D., D.M., B.Me., K.P., M.R., C.G.K., and M.M.B) and GPCR Collaborative (J.D., V.T., and M.M.B) funding mechanisms within St Jude. We thank the domain experts, Drs. Balaji Santhanam, Katarina Nemec, Preethi Badrinarayan, Ben Leslie, Kalyan Immadisetty, and Benjamin Lang for their assistance in creating the Q&A datasets used for benchmarking the biomedical literature and public datasets, Jino John for helping with evaluation of the kinase literature, and Nancy Gough and Jeyanthy Eswaran for their comments on the evaluation. We gratefully acknowledge Andrea Williams, Marcia Sabino de Freitas, Heather Fosburgh, Patricia Loggetto and Melisssa Santiago together with D.B.A. and C.G.L. from St. Jude Global for their evaluation of health policies.We thank Pranay Kumar Anugu, Roshan Shrestha, and Piby Paul for their help with deploying MyGPT on the VMs approved for PHI data. We also thank Drs. Jim Downing, Paul Taylor, Elly Park, Babu group and Blue-Sky Kinase team for comments on this work as well as Ashok Kumar Reddy Boddu, Ed Suh, Michael Brewer, Darrell Gentry, Chad Burdyshaw, and Keith Perry from the Information Services department, and Andrew Dedmon, Valentina Perez, and Brian Elrod from the Information Security team of St. Jude for their support.

Author Contributions
J.P. and M.M.B. conceived the MyGPT concept and all confidence metrics. J.P. and J.D. contributed equally to designing a prototype for MyGPT software. J.P. developed the MyGPT software and pipeline. B.Me., V.T., and D.M. helped with multiple rounds of manual evaluation of the benchmark datasets. D.M. reviewed the code. H.F. implemented the recursive chunking approach and evaluated MyGPT and PrivateGPT on public datasets. T.Y. evaluated MyGPT, Dify, and Kotaemon on public datasets and ran evaluation pipelines for MyGPT on health policies and the kinase literature. J.D. collected open RAG Benchmark datasets and set up an evaluation pipeline for MyGPT. Cai L. provided methods and guidance for sample size calculations for RAG applications; R.U. and J.P. verified their use with MyGPT and other RAG solutions. R.U. and J.P. conducted a cost analysis of MyGPT and other similar solutions. S.P. designed a visual settings menu for uploading and customizing chat settings in the MyGPT interface. With the guidance of C.G.K., K.P., M.R., and B.Me. collected Q&A datasets and performed validation of MyGPT answers for the kinase Q&As. D.B.A. and C.G.L. defined and collected Q&A datasets and coordinated and performed validated MyGPT answers for health policies with the St. Jude Health Systems team, facilitated by C.G.L. and C.R.G. C.M. collected Q&A datasets and performed validation of MyGPT answers for BMTCT SOPs with the guidance of D.K. and S.G. S.M.D.A. assisted in analyzing confidence metric values and created a random forest model for Hallucination Index. S.A. installed and supported Ollama on a server with an A100 GPU, D.A. set the automation pipeline and GitHub repository management, A.W. implemented frontend unit tests, and A.W. and D.A. set up cloud Azure VMs. J.P., I.C., M.S., B.Mo., and M.M.B. wrote the manuscript. J.P. led the project. M.M.B. managed the project and set its overall vision and specific directions.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information is available for this paper.

Correspondence and request for materials should be addressed to Jaimin Patel or M. Madan Babu.

Code availability
All code, data, and tools are available at https://github.com/stjude-c3d/MyGPT. [This link is currently private and will be made public upon acceptance for publication. The source code is provided as a zip file in Supplementary Information for the review process. We have also provided a URL with username and password for each reviewer to the journal editor; please contact the editor if you prefer to evaluate the

hosted version of MyGPT]. We performed vulnerability testing using Snyk to evaluate best practices in software development to eliminate security weaknesses. Our code and data are licensed under GPL-3.0 license, while LLM models and LLM tools used for the project are available under the licenses of the respective sources.

SUPPLEMENTARY TABLES

Supplementary Table 1: Example LLMs that can be used with MyGPT via Ollama.

LLM	Parameters	Size	Developed by	Note	Ideal hosting environment	Minimum requirements
Llama2:7b	6.7B	3.8 GB	Meta	Released on 18th July 2023	PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Llama3:8b	8.03B	4.7 GB			PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Llama3:70b	70.6B	40 GB			VM, server, Cloud service	1 GPU, 64 GB RAM
Llama3.1:8b	8.03B	4.9 GB		Released on 23rd July, 2024	PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Llama3.2:1b	1.24B	1.3 GB		Released on 25th September, 2024	Edge AI device, PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Llama3.3:70b	70.6B	43 GB		Released on 7th December, 2024	VM, server, Cloud service	1 GPU, 64 GB RAM
Gemma:7b	8.54B	5 GB	Google DeepMind	Released on 21st February 2024	PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Gemma2:9b	9.24B	5.4 GB		Released on 27th June 2024	PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Gemma2:2b	2.61B	1.6 GB			Edge AI device, PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Gemma3:1b, 4b	1B, 4.3B	0.8GB, 3.3GB		Released on 12th March 2025	Edge AI device, PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Gemma3:12b, 27b	12.2B, 27.4B	8.1GB, 17GB		Released on 12th March 2025	PC/Laptop, Cloud service	1 GPU, 32 GB RAM
Gemma4:26b, 31b	25.8B, 31.3B	18GB, 20GB		Released on 2nd April 2026	PC/Laptop, Cloud service	1 GPU, 32 GB RAM
Gpt-oss:20b	20.9B	14GB	OpenAI	Released on 5th August 2025 (Thinking model)	PC/Laptop, Cloud service	1 GPU, 32 GB RAM
DeepSeek-R1:8b, 14b	8.19B, 14.8B	5.2GB, 9GB	DeepSeek	Released on January 20, 2025 (Thinking model)	PC/Laptop, Cloud service	1 GPU, 32 GB RAM
Qwen3:8b, 14b	8.19B, 14.8B	5.2GB, 9.3GB	Alibaba	Released on April 29th, 2025 (Thinking model)	PC/Laptop, Cloud service	1 GPU, 16 GB RAM
Mistral:7b	7.2B	4.1 GB	Mistral AI	Released on 27th September 2023	PC/Laptop, Cloud service	1 GPU, 8 GB RAM
Vicuna:7b	6.74B	3.8 GB	LMSYS Org, UC Berkeley	Finetuned Llama2	PC/Laptop, Cloud service	1 GPU, 8 GB RAM

Supplementary Table 2: LLM performance for different environments.

LLM size category	LLM	Host environment	Infrastructure	GPU	Avg. Speed (seconds)
Mid-size LLMs (most common)	**Llama2:7b**	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	3.80
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	6.46
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	2.89
		Apple Mac m1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	9.26
		Apple Mac m3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	5.21
	Llama3:8b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	3.34
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	4.96
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	3.40
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	7.47
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	6.00
	Llama3.1:8b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	2.49
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	6.31
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	2.05
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	7.05
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	4.45
	Gemma:7b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	2.74
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	4.38
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	1.61
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	5.52
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	2.80
	Gemma2:9b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	2.72
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	4.57
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	1.46
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	6.01
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	3.28
	Mistral:7b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	3.37
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	5.69
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	2.67
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	7.87

		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	5.28
	Vicuna:7b	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	3.96
		Google Cloud Platform	2 vCPUs, 12 GB RAM	1 Nvidia T4 Tensor Core	8.73
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	3.56
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	9.51
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	8.93
Small LLMs	**Llama3.2:3b**	Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	3.79
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	2.28
		Nvidia Orin-nano	1 CPU, 8 GB RAM	1 1024-core NVIDIA Ampere	19.88
	Llama3.2:1b	Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	2.15
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	1.52
		Nvidia Orin-nano	1 CPU, 8 GB RAM	1 1024-core NVIDIA Ampere	9.82
	Gemma:2b	Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	1.25
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	0.80
		Nvidia Orin-nano	1 CPU, 8 GB RAM	1 1024-core NVIDIA Ampere	5.26
Large LLMs	**Llama3:70b**	Google Cloud Platform	2 vCPUs, 83 GB RAM	1 Nvidia A100 Tensor Core	11.17
		St. Jude internal server	64 CPUs, 2 TB RAM	1 Nvidia A100 Tensor Core	31.11
		Apple Mac M1	8 CPUs, 64 GB RAM	1 Apple M1 Metal	60.27
		Apple Mac M3	8 CPUs, 64 GB RAM	1 Apple M3 Metal	49.53

Supplementary Table 3. Cost of hosting LLMs on Cloud infrastructure.

Cloud service provider	Infrastructure	GPU	Avg. cost per month (USD)
Azure	24 vCPUs, 220 GB RAM	1 Nvidia A100 Tensor Core	$2,681.29
Azure	8 vCPUs, 56 GB RAM	1 Nvidia T4 Tensor Core	$548.96
Google	12 vCPUs, 85 GB RAM	1 Nvidia A100 Tensor Core	$2,682.57
Google	8 vCPUs, 30 GB RAM	1 Nvidia T4 Tensor Core	$374.03
Google	4 vCPUs, 16 GB RAM	1 Nvidia L4 Tensor Core	$516.99

Supplementary Table 4. Benchmarking of various MyGPT steps for different library sizes.

Device	Pages	PDF count	Chunks count (fixed chunk size)	Upload time	Retrieval time (1st)	Retrieval time (avg)	Generation time (1st) llama3:7b	Generation time (avg)	end-to-end time (avg)
Mac Laptop (M1 chip)	10	1	70	2m10s	1.28s	1.06s	6.39s	2.99s	11s
	100	2	235	2m33s	1.18s	0.62s	4.16s	6.03s	17s
	1000	4	3211	5m53s	0.73s	0.59s	3.16s	3.9s	12s
	10000	791	58869	95m19s	1.94s	0.64s	5.52s	5.82s	17s
Mac Laptop	10	1	70	2.05s	0.38s	0.225s	0.84s	1.205s	4.5s
	100	2	235	4.9s	0.1s	0.0775s	2.04s	2.295s	4.5s

(M4 chip)	1000	4	3211	1m18s	0.14s	0.11333s	1.9s	2.049s	4.2s
	10000	791	58869	23m51s	0.69s	0.651s	1.86s	2.124s	5.0s
SJ VM	10	1	70	11s	5.63s	0.80s	11.22s	0.51s	11s
	100	2	235	13s	0.77s	0.79s	1.2s	1.07s	6.75s
	1000	4	3211	2m12s	0.68s	0.73s	1.16s	1.49s	12.6s
	10000	791	58869	46m2s	0.77s	0.85s	1.19s	1.39s	8s

Supplementary Table 5: Sample size calculations for MyGPT vs other RAG solutions.

Dataset	MyGPT correction rate (p_1)	Other RAG Tools	Correction rate ($p2$)	MyGPT-vs-RAG-tool corrections correlation (ρ)	Required number of docs (c)
PubMedQA Q&A=294, docs used=294)	0.8163	RAGFlow	0.7755	0.608	806
		PrivateGPT	0.5102	0.062	43
		Dify	0.6803	0.146	179
		Kotaemon	0.4558	0.134	28
BioASQ (Q&A=163, docs used=210)	0.9141	RAGFlow	0.9018	0.561	5106
		PrivateGPT	0.7485	0.327	72
		Dify	0.6564	0.009	48
		Kotaemon	0.7607	0.085	108
Open RAG Benchmark (Q&A=170, docs used =165)	0.9294	RAGFlow	0.9	0.444	1063
		PrivateGPT	0.7824	0.299	84
		Dify	0.7882	0.306	89
		Kotaemon	0.5588	0.218	20

Supplementary Table 6: Cost of hosting MyGPT on commodity hardware for group or department usage.

Hardware	Configuration	GPU	Price	Best LLM to host
Mac Mini	64 GB RAM, 512 GB Storage	M4 Pro chip	$1,839	Gpt-os:20b, Gemma4:31b
Framework Desktop	64 GB RAM, 512 GB Storage	AMD Ryzen™ AI Max+ 395	$2,213	Gpt-os:20b, Gemma4:31b
Mac Studio	128 GB, 512 GB storage	M4 Max chip	$3,149	Gpt-oss:120b, Llama3.3:70b
Framework Desktop	64 GB RAM, 512 GB Storage	AMD Ryzen™ AI Max+ 395	$3,253	Gpt-oss:120b, Llama3.3:70b
Nvidia DGX Spark	128 GB, 1 TB storage	Nvidia Blackwell Architecture	$4,000	Gpt-oss:120b, Llama3.3:70b

Supplementary Table 7: Cost comparison with various commercial and open-source RAG solutions.

Tool	Type	Pricing structure	Cost	Notes	Sources
Microsoft 365 Copilot (Business)	Commercial, SaaS	Per user/month	$18/user/mo (through Jun 2026), then $21/user/mo	Requires a qualifying M365 license on top of the Copilot fee. RAG grounded in M365 Graph data.	https://copilot-experts.com/microsoft-copilot-pricing-guide/
Microsoft 365 Copilot (Enterprise)	Commercial, SaaS	Per user/month	$30/user/mo on top of M365 E3 ($39) or E5 ($60)	True all-in cost ranges from $66 - $87/user/month when base license is included.	https://redresscompliance.com/microsoft-copilot-licensing-guide-2026.html, https://www.directionsonmicrosoft.com/microsoft-to-increase-office-suite-prices-across-the-board-starting-july-2026/

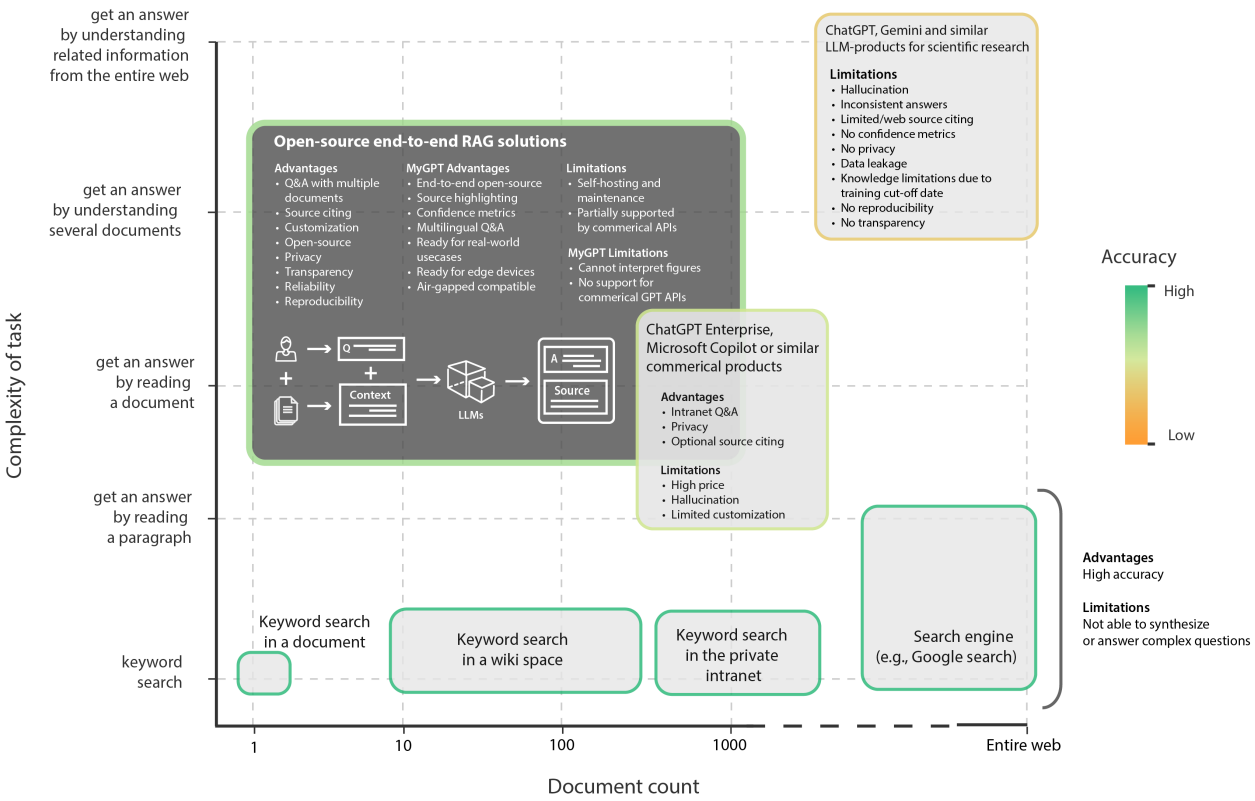
Gemini for Google Workspace	Commercial, SaaS	Per user/month	Now bundled into Workspace plans ($7--$22/user/mo after 17--22% price increase)	Previously sold as a $20 - $30/user/mo add-on; standalone Gemini add-ons discontinued for new purchases. Vertex AI API access is separate.	https://www.lineserve.net/blog/google-workspace-pricing-2026, https://www.cumulusglobal.com/google-workspace-gemini-ai-features-and-pricing-changes/
Gemini Enterprise (Google Cloud)	Commercial, SaaS	Per user/month + consump-tion	$30/user/mo as Workspace add-on for existing subscribers; volume discounts at 500+ and 1000+ seats. Vertex AI: $1.88/M input tokens, $7.50/M output tokens.	Includes agentic platform, compliance controls (HIPAA, FedRAMP).	https://atonementlicensing.com/blog/gemini-enterprise/, https://workspace.google.com/pricing
ChatGPT Business	Commercial, SaaS	Per user/month	$25/user/mo (annual) $30/user/mo (monthly)	Includes admin controls, SOC 2, SAML SSO. Data not used for training.	https://chatgpt.com/pricing/, https://tldv.io/blog/chatgpt-pricing/
ChatGPT Enterprise	Commercial, SaaS	Custom quote	Negotiated	Adds custom data retention, volume licensing, dedicated support.	https://openai.com/business/chatgpt-pricing/
Glean	Commercial, SaaS	Per user/month (quote-based)	$45-$50+ user/mo base; AI add-on ~$15/user/mo extra	100-seat minimum; enterprise contracts often exceed $200K/year. No public pricing. Renewal increases of 7--12% reported.	https://fritz.ai/glean-review/, https://www.gosearch.ai/blog/glean-pricing-explained/
Vectara	Commercial, SaaS/VPC	Usage-based (credits)	Enterprise plans: Small $100K/yr, Medium $250K/yr, Large $500K/yr	30-day free trial. Pricing based on index size and query volume, not per-seat. VPC and on-prem options available.	https://www.vectara.com/pricing, https://www.extruct.ai/hub/vectara-com-funding/
PrivateGPT (Abstracta)	Commercial, self-hosted (AWS)	Custom quote	No public list price; request private offer via AWS Marketplace	Self-hosted on EKS/ECS. Includes SSO, per-user spend caps, RAG pipelines, usage analytics.	https://aws.amazon.com/marketplace/pp/prodview-j3dxmeroxeqju
Kotaemon	Open source (Apache 2.0)	Free (self-hosted)	$0 (software)	Self-hosted via Docker. Supports OpenAI, Azure, Ollama, local LLMs. Ongoing costs are infra + LLM API tokens.	https://github.com/Cinnamon/kotaemon, https://www.marktechpost.com/2024/09/01/kotaemon-an-open-source-rag-based-tool-for-chatting-with-your-documents/
LlamaIndex / LangChain	Open source	Free	$0 (framework)	Developer frameworks, not turnkey products. Total cost depends on hosting, vector DB, and LLM API spend.	https://www.meilisearch.com/blog/rag-tools
Meilisearch	Open source + Cloud	Freemium / usage-based	Build plan $30/mo (50K searches, 100K docs);	Search engine with RAG integration, not a full copilot/assistant.	https://www.meilisearch.com/blog/rag-tools

			self-hosted is free		

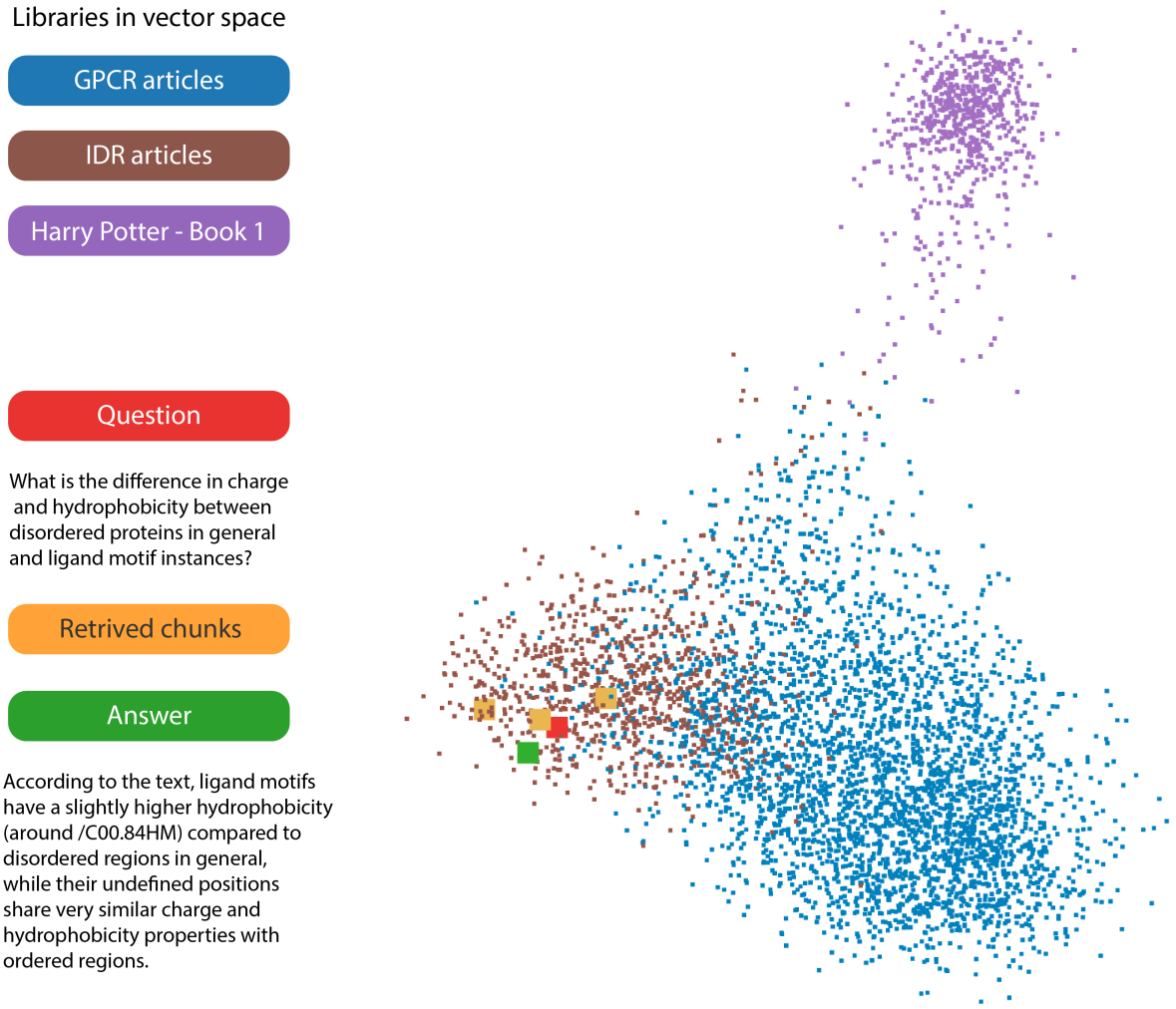
Key takeaways

- Per-user tools carry hidden costs: Microsoft Copilot requires a qualifying M365 base license, so the real per-seat cost is roughly double the headline add-on price. Glean layers on support fees (~10% of contract value) and reported renewal increases.
- Open-source options shift cost to operations. Kotaemon, LlamaIndex, and LangChain have zero licensing fees but require engineering time, infrastructure, and LLM API bills.
- Gemini pricing restructured in 2025. Google folded Gemini into base Workspace plans with a 17--22% price increase, eliminating the separate add-on SKU for new customers. Existing subscribers on legacy Gemini add-ons should evaluate whether they are double-paying for capabilities also available through Vertex AI.
- Vectara is the outlier in pricing model. It charges by usage credits (index size + queries) rather than seats, which can benefit organizations with many documents but relatively few heavy users.

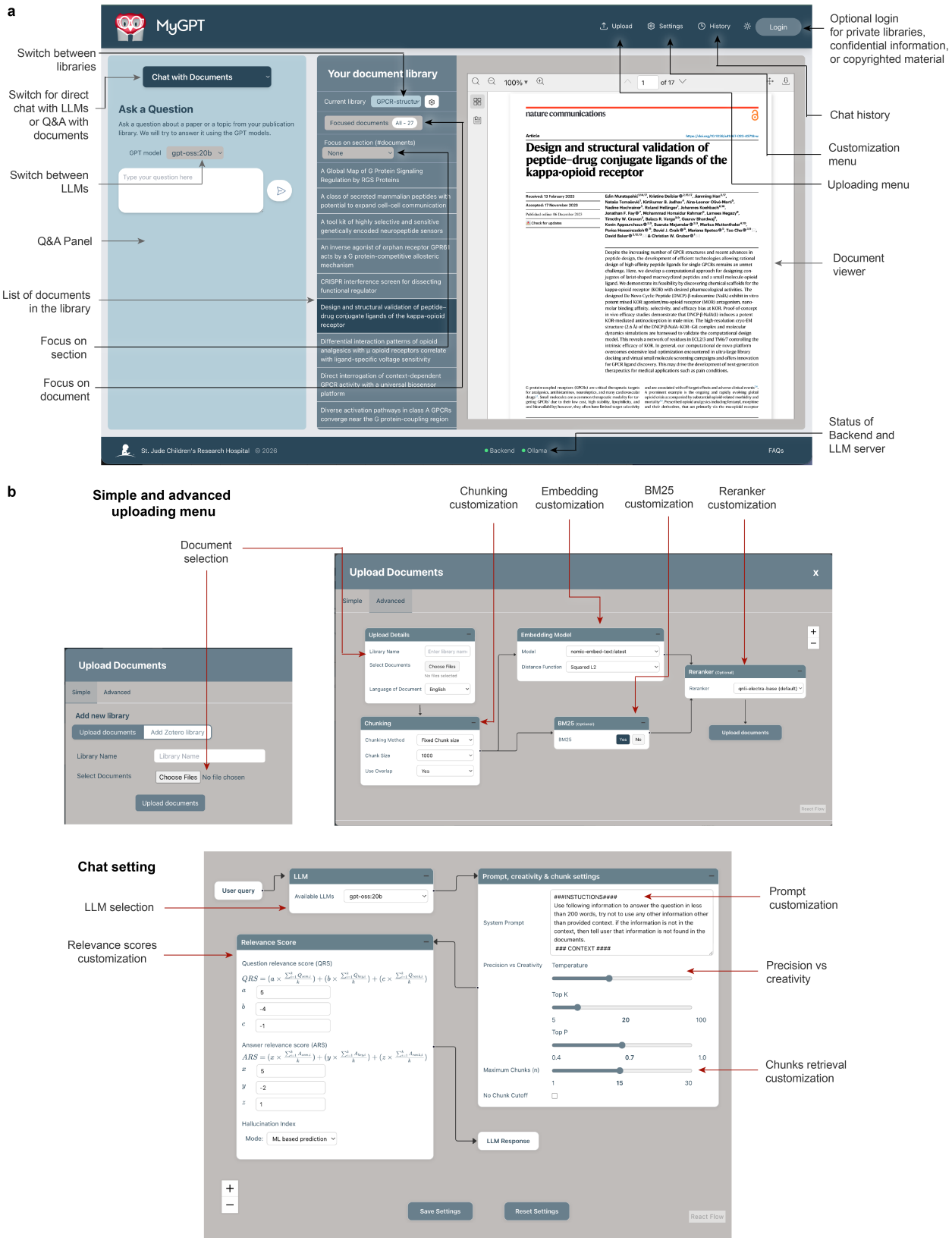
Supplementary Figures



Supplementary Figure 1. **Information-seeking tasks and solutions**. Keyword searches from a single document, multiple documents, a private intranet, or the entire web are routine tasks for most knowledge workers that have been optimized over several decades to provide accurate information. With ChatGPT and similar products, users can get answers to complex questions requiring processing of related material, but accuracy and factuality are lower than those of typical search solutions. MyGPT and other RAG solutions fill a void in the landscape of available discovery tools. The orange-to-green color of the framed boxes indicates the level of accuracy of the information obtained from the indicated task.

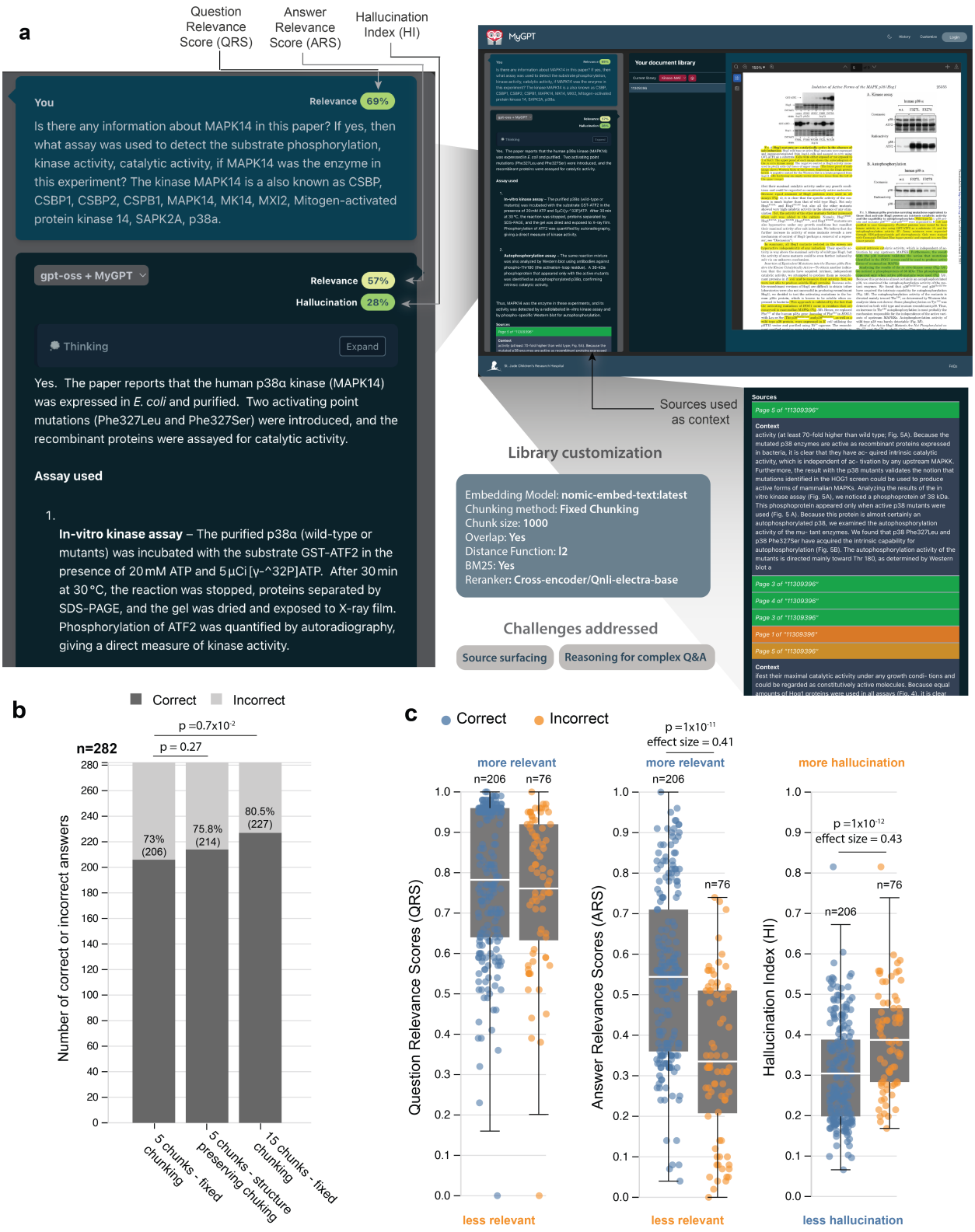


Supplementary Figure 2. Visualization of chunks from library of documents in the vector space of embeddings. The PCA plot visualizes vector embeddings of chunks from two different libraries of publicly available biomedical articles on (a) G protein-coupled receptors (GPCRs; 27 articles) in blue, (b) intrinsically disordered regions (IDRs; 8 articles) in brown, and text from (c) Harry Potter-Book 1 in purple (**Supplementary Data**). Each dot in the plot denotes a chunk from the three different libraries. A question related to the IDR library is converted into a vector (red square). MyGPT then identifies the chunks (orange squares) that are closest to the question and provides them as context for the LLM to generate a response. The generated answer (green square), when converted into a vector using the same embedding model, is closer to the context, highlighting similarity.



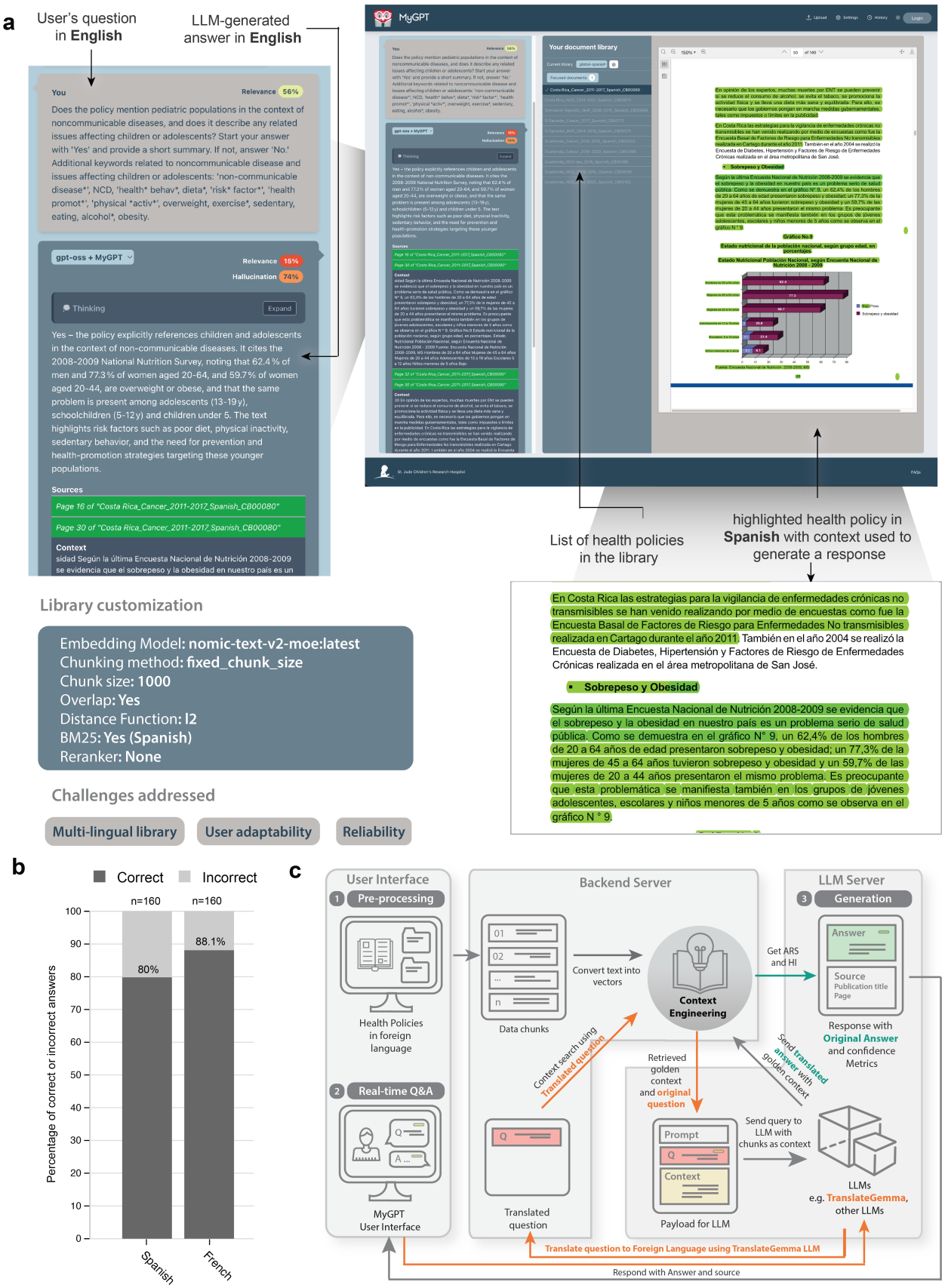
Supplementary Figure 3. **User interface for MyGPT a** MyGPT UI home screen. The home screen displays a chat box (left) with options to switch between pre-loaded document libraries and LLMs, and a dropdown menu that lets users chat directly with the LLM or with documents. The “Your document library” panel (middle) displays a list of documents in the library, with the selected document loaded and visible in

the document viewer (right). The top navigation bar displays icons to enable user login to private libraries (e.g., group- or department-level access). Users can also check chat history for various libraries and access the settings menu from this area. Users can verify whether the Backend and LLM server are properly connected and functional from the page footer. **b** MyGPT upload and chat menu. From the Upload menu, users can upload documents and create a library; each preprocessed library is accessible from the MyGPT home page. When uploading, users can either select a Simple upload with default customization pre-selected or can go to Advanced settings to customize various components such as chunk size, chunk method, chunk overlap, embedding model, distance function, BM25, and Reranker. In the chat settings, users can change the LLM, prompt, precision vs. creativity parameters, chunk retrieval customization, and relevance score customization.

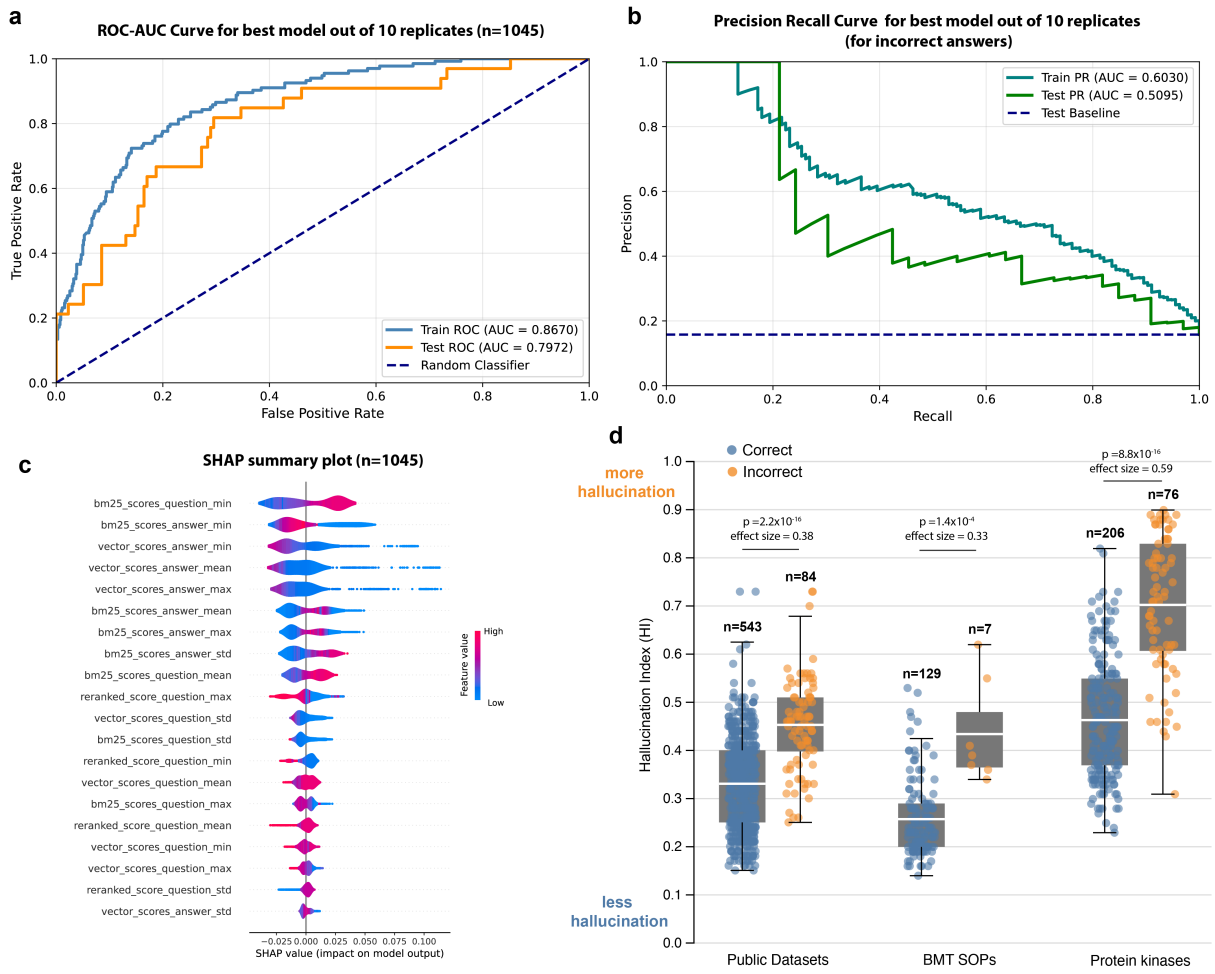


Supplementary Figure 4. **Use case 1: Q&A with large library of biomedical literature. a** Example of a complex question and answer for the library of Biomedical literature on protein kinases. The user's question is shown along with the highlighted section of the articles used by the LLM (gpt-oss:20b) to generate the answer. This use case addresses the challenge of retrieving related information from multiple sections of an article and synthesizing an answer after analyzing the data. The library was created using the nomic-

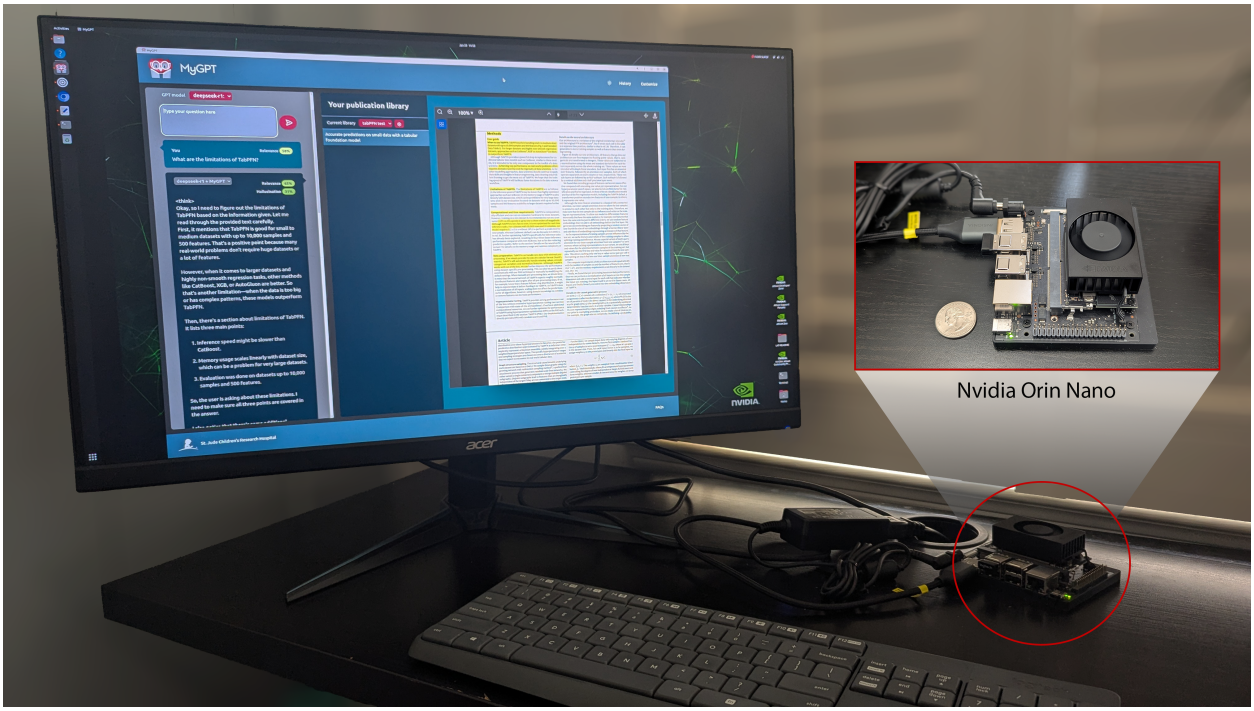
embed-text embedding model, which utilizes the reasoning capabilities of the LLM (gpt-oss:20b) to generate an answer. **b** 282 questions (with ground truth) were collected from 83 documents for this evaluation. MyGPT was able to correctly answer 206 questions with 73% accuracy. The same experiment performed with 15 chunks per question (n=15, maximum chunks per semantic and keyword search) rather than the default 5 chunks yielded more correct answers with 80.13% (238/297) accuracy, showing significant improvement (p = $0.7x10^{-2}$; effect size = 0.34 (Φ, medium); McNemar chi-square test). Also, changing the chunking strategy from fixed to structure-preserving (with n=5) yielded 75.8% (214/282) accuracy, which did not provide a significant improvement (p = 0.27; effect size=0.15 (Φ, small); McNemar chi-square test) over fixed chunking with 5 chunks. **c** MyGPT's Question relevance score (QRS) distribution for questions with correct or incorrect answers (p = 0.13; effect size = 0.08 (r, small); Mann Whitney U test) is not statistically different, but the Answer relevancy score (ARS) distribution for correct and incorrect answers (p = $5x10^{-12}$; effect size = 0.41 (r, medium); Mann Whitney U test) shows higher scores for the correct answers. MyGPT's Default Hallucination Index (HI) for correct and incorrect answers (p = $6x10^{-13}$; effect size = 0.43 (r, medium); Mann Whitney U test) shows significantly higher hallucination index values for incorrect answers.


a
User's question in English
LLM-generated answer in English
You
Relevance 56%
Does the policy mention pediatric populations in the context of noncommunicable diseases, and does it describe any related issues affecting children or adolescents? Start your answer with 'Yes' and provide a short summary. If not, answer 'No.' Additional keywords related to noncommunicable disease and issues affecting children or adolescents: 'non-communicable disease*', NCD, 'health* behav*, dieta*, 'risk* factor*', 'health promot*', 'physical *activ*', overweight, exercise*, sedentary, eating, alcohol*, obesity.
gpt-oss + MyGPT
Relevance 15%
Hallucination 74%
Thinking
Expand
Yes – the policy explicitly references children and adolescents in the context of non-communicable diseases. It cites the 2008-2009 National Nutrition Survey, noting that 62.4 % of men and 77.3 % of women aged 20-64, and 59.7 % of women aged 20-44, are overweight or obese, and that the same problem is present among adolescents (13-19 y), schoolchildren (5-12 y) and children under 5. The text highlights risk factors such as poor diet, physical inactivity, sedentary behavior, and the need for prevention and health-promotion strategies targeting these younger populations.
Sources
Page 16 of "Costa Rica_Cancer_2011-2017_Spanish_CB00080"
Page 30 of "Costa Rica_Cancer_2011-2017_Spanish_CB00080"
Context
sidad Según la última Encuesta Nacional de Nutrición 2008-2009 se evidencia que el sobrepeso y la obesidad en nuestro país es un
MyGPT
Your document library
List of health policies in the library
highlighted health policy in Spanish with context used to generate a response
En Costa Rica las estrategias para la vigilancia de enfermedades crónicas no transmisibles se han venido realizando por medio de encuestas como fue la Encuesta Basal de Factores de Riesgo para Enfermedades No transmisibles realizada en Cartago durante el año 2011. También en el año 2004 se realizó la Encuesta de Diabetes, Hipertensión y Factores de Riesgo de Enfermedades Crónicas realizada en el área metropolitana de San José.
Sobrepeso y Obesidad
Según la última Encuesta Nacional de Nutrición 2008-2009 se evidencia que el sobrepeso y la obesidad en nuestro país es un problema serio de salud pública. Como se demuestra en el gráfico N° 9, un 62,4% de los hombres de 20 a 64 años de edad presentaron sobrepeso y obesidad; un 77,3% de la mujeres de 45 a 64 años tuvieron sobrepeso y obesidad y un 59,7% de las mujeres de 20 a 44 años presentaron el mismo problema. Es preocupante que esta problemática se manifiesta también en los grupos de jóvenes adolescentes, escolares y niños menores de 5 años como se observa en el gráfico N ° 9.
Library customization
Embedding Model: nomic-text-v2-moe:latest
Chunking method: fixed_chunk_size
Chunk size: 1000
Overlap: Yes
Distance Function: l2
BM25: Yes (Spanish)
Reranker: None
Challenges addressed
Multi-lingual library
User adaptability
Reliability
b
Correct
Incorrect
n=160
n=160
80%
88.1%
Percentage of correct or incorrect answers
Spanish
French
c
User Interface
1 Pre-processing
Health Policies in foreign language
2 Real-time Q&A
MyGPT User Interface
Backend Server
Data chunks
Convert text into vectors
Context Engineering
Context search using Translated question
Retrieved golden context and original question
Translated question
Payload for LLM
LLM Server
3 Generation
Get ARS and HI
Answer
Source
Publication title
Page
Response with Original Answer and confidence Metrics
Send translated answer with golden context
Send query to LLM with chunks as context
Prompt
Q
Context
LLMs
e.g. TranslateGemma, other LLMs
Translate question to Foreign Language using TranslateGemma LLM
Respond with Answer and source

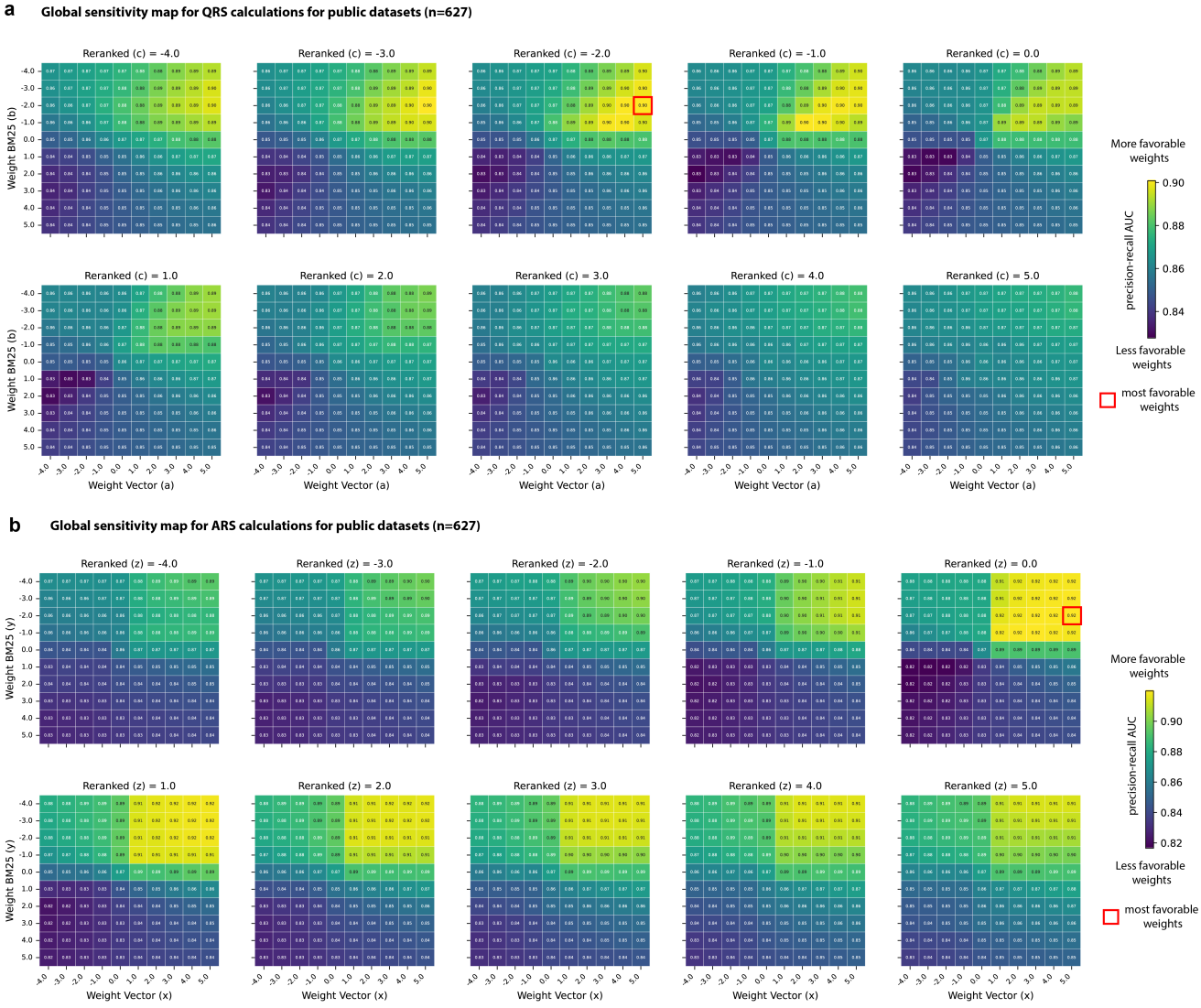
Supplementary Figure 5. **Use case 2: Global health policies in multiple languages. a** Example of a question and answer for global health policies with documents in Spanish from the Policy Monitor, a global set of national cancer-related Health Policies library curated by St. Jude Gobal. A user's question, posed in English, was compared across various document sections using an embedding model with multilingual support (nomic-text-v2-moe:latest). The context in Spanish was utilized by the language model (GPT-oss:20b) to generate the answer in English. **b** 128 and 160 questions (with ground truth) were collected from 10 health policies in Spanish or French, respectively, providing 320 total Q&A for this evaluation. MyGPT correctly answered 110 (80%) of questions in Spanish and 141 (88.1%) of questions in French from the library. **c** MyGPT uses the user-provided document language to select the appropriate embedding model and BM25 version. User questions were translated into the document's language using the TranslateGemma LLM before performing semantic and keyword searches for context retrieval. Using this context, the user-selected LLM generates a response in the user's language, and MyGPT uses translated answers from TranslateGemma to compute ARS and HI. Modifications to MyGPT's workflow for this use case are indicated in orange (backend) or green (LLM) bold font.



Supplementary Figure 6. **Random forest model for generalizable HI predictor. a** Performance of 10 replicates of the random forest model trained and tested on 1045 Q&A from public and private datasets with an 80:20 split to predict incorrect answers. For the best perfoming replicate out of 10, the ROC-AUC is 86.70% for training and 79.72% for testing. **b** The precision-recall curve of the same best perfroming replicate of the model shows 60.3% for training and 50.95% for testing. The baseline of 17% represents the proportion of incorrect answers for 1045 samples, which will be the performance of a random predictor. Precision-recall of 50.95% for the random forest model signifies a 3-fold better performance than that of a random predictor. **c** SHAP summary plot of 20 features used to train and test the random forest model to predict incorrect answers. Features are ranked by mean absolute SHAP value, with positive SHAP values pushing the model toward predicting “incorrect” and negative SHAP values toward predicting “correct” answers. The high feature values are marked in pink and the low feature values in blue. **d** The HI values of all 1045 Q&A using the random forest model. The 3 public datasets combined (p = 2.2×10^{-16}; effect size = 0.38 (r, medium); Mann Whitney U test), BMT (Bone Marrow Transplantation) SOPs (p = 1.4×10^{-4}; effect size = 0.33 (r, medium); Mann Whitney U test) and Protein kinases (p = 8.8×10^{-16}; effect size = 0.59 (r, large); Mann Whitney U test) datasets display significant differences in HI scores between correct and incorrect Q&A pairs with large effect size.



Supplementary Figure 7. MyGPT running on Nvidia Orin Nano with small-size LLMs. Small LLMs include Llama3.2:3b, Llama3.2:1b, Gemma2:2b, and DeepSeek-R1:1.5b that are ideal for remote locations with limited or no internet access or for air-gapped deployments.



Supplementary Figure 8. Global sensitivity analysis to calculate QRS and ARS for public datasets. **a** Heatmaps showing classification performance (AUC) across combinations of weight parameters used in QRS computation for a range of vector, BM25, and reranking weight cutoffs. The weights marked in red box are the optimal weights for QRS calculations, 5 for vector, -2 for BM25 and -2 for Reranker. **b** Corresponding sensitivity maps for ARS across weight configurations for a range of vector, BM25, and reranking weight cutoffs. Color scale indicates AUC, highlighting the robustness and dependence of QRS and ARS on parameter choices. The weights marked in red box are the optimal weights for ARS calculations, 5 for vector, -2 for BM25 and 0 for Reranker.