CollEX – A Multimodal Agentic RAG System Enabling Interactive Exploration of Scientific Collections

Florian Schneider[†], **Narges Baba Ahmadi**[† *], **Niloufar Baba Ahmadi**[† *]
Iris Vogel[‡], **Martin Semmann**[†], **Chris Biemann**[†]

[†]Hub of Computing and Data Science
[‡]Center for Sustainable Research Data Management
University of Hamburg, Germany

Correspondence: florian.schneider-1@uni-hamburg.de

[*]Equal contributions, sorted alphabetically.

Abstract

In this paper, we introduce CollEX, an innovative multimodal agentic Retrieval-Augmented Generation (RAG) system designed to enhance interactive exploration of extensive scientific collections. Given the overwhelming volume and inherent complexity of scientific collections, conventional search systems often lack necessary intuitiveness and interactivity, presenting substantial barriers for learners, educators, and researchers. CollEX addresses these limitations by employing state-of-the-art Large Vision-Language Models (LVLMs) as multimodal agents accessible through an intuitive chat interface. By abstracting complex interactions via specialized agents equipped with advanced tools, CollEX facilitates curiosity-driven exploration, significantly simplifying access to diverse scientific collections and records therein. Our system integrates textual and visual modalities, supporting educational scenarios that are helpful for teachers, pupils, students, and researchers by fostering independent exploration as well as scientific excitement and curiosity. Furthermore, CollEX serves the research community by discovering interdisciplinary connections and complementing visual data. We illustrate the effectiveness of our system through a proof-of-concept application containing over 64,000 unique records across 32 collections from a local scientific collection from a public university.

1 Introduction

The exploration of scientific knowledge is a cornerstone of human progress. However, the vast and rapidly growing body of scientific literature presents significant challenges for educators and learners, who often find themselves overwhelmed by the sheer volume and complexity of information. Despite advancements in information retrieval and knowledge discovery (Santhanam et al., 2022; Zhu et al., 2023; Li et al., 2024b), existing search systems for rich and complex data often lack the interactivity, intuitiveness, and cross-modal search capabilities (Faysse et al., 2024; Zhai et al., 2023; Zhao et al., 2023b) to engage diverse audiences, such as students, teachers, or researchers. This limitation negatively affects educational settings where fostering curiosity is essential.

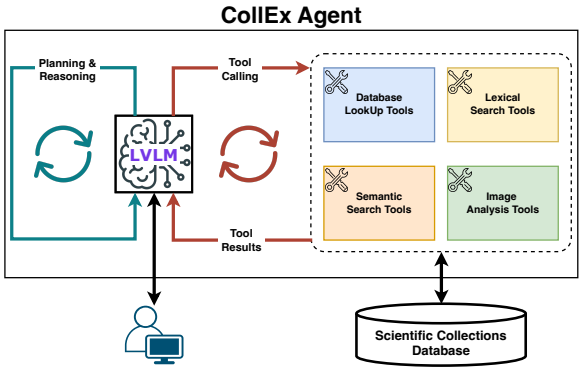


Figure 1: An overview of the CollEX Agentic System.

With this paper, we introduce CollEX, a multimodal agentic Retrieval-Augmented Generation (RAG) system (Lewis et al., 2020; Zhao et al., 2023a; Xie et al., 2024) and reimagine how users explore and interact with scientific collections such as those collected and managed by the Smithsonian Institution[1] or local collections from public universities. CollEX uses state-of-the-art Large Vision-Language Models (LVLMs)(Liu et al., 2023; Team et al., 2023; Hurst et al., 2024; Yang et al., 2024; Team et al., 2025) as multimodal agents (Xie et al., 2024; Wang et al., 2024) through an intuitive chat interface. Unlike traditional systems requiring expert knowledge, CollEX promotes curiosity-driven exploration, simplifying access and increasing engagement.

The core of CollEX is its multimodal agentic RAG system, which abstracts complex interactions using specialist agents equipped with various tools (Patil et al., 2024). This simplifies the explo-

[1]https://www.si.edu/collections

Proceedings of the 1st Workshop on Multimodal Augmented Generation via Multimodal Retrieval (MAGMaR 2025), pages 18–39

ration of extensive scientific collections, catering to users with diverse backgrounds and expertise, thereby overcoming accessibility issues (Achiam and Marandino, 2014). The system integrates texts and images, offering intuitive access to scientific concepts.

CollEX is especially beneficial in education, fostering curiosity and engagement. For instance, teachers can get inspiration to prepare visually rich lessons, retrieve relevant information, and facilitate interactive assignments. Pupils can independently explore the collections, transforming static materials into dynamic learning experiences. Moreover, CollEX supports higher education by encouraging independent exploration and enhancing critical thinking skills.

Beyond education, CollEX aids researchers in discovering interdisciplinary connections, eventual related work, or visual data complements. It autonomously enriches search queries, facilitating easier contextualization and increasing accessibility to scientific collections, thereby supporting national and international scientific connectivity (Weber, 2018).

This paper introduces CollEX's general system architecture[2]and inner workings, combining state-of-the-art LVLMs, advanced prompting and RAG techniques, cross-modal search, and agentic reasoning and planning.

Moreover, we provide three exemplary user stories to demonstrate the system by implementing a proof-of-concept application to explore 32 diverse scientific collections comprising over 64,000 unique items.

2 Related Work

2.1 Cross-Modal Information Retrieval

Cross-modal information retrieval powered by multimodal embeddings is the key foundation for systems navigating or exploring textual and visual data such as CollEX. Recent developments in multimodal embedding models (Tschannen et al., 2025) that compute semantically rich dense vector representations in an aligned vector space for texts and images, have significantly improved over the popular text-image encoder model, commonly known as CLIP (Radford et al., 2021). This progress was primarily driven by billion-scale high-quality text-image datasets (Schuhmann et al., 2022), improvements in architecture and training regimes (Zhai et al., 2023), and improved Vision Transformers (Alabdulmohsin et al., 2023) Despite their applications in "pure" information retrieval settings, the image encoders of the multimodal embedding models also play a crucial role in the advancement of Large Vision Language Models (LVLMs) (Liu et al., 2023; Yang et al., 2024; Geigle et al., 2025) as they are often used to compute the visual tokens processed by the LVLMs.

2.2 Multimodal Retrieval Augmented Generation

Multimodal RAG (Zhao et al., 2023b) systems integrate various knowledge formats, including images, code, structured databases, audio, and video, to enhance the knowledge of LVLMs at inference time. Zhao et al. (2023b) further highlight that such multimodal data helps mitigate hallucinations and improve interpretability and reasoning by grounding responses in diverse multimodal information. Riedler and Langer (2024) demonstrate the advantages of incorporating images into textual retrieval systems within industrial applications. Their findings suggest that image-derived textual summaries often outperform purely embedding-based multimodal approaches.

2.3 Agentic RAG

As described above, traditional RAG systems combine LLMs' or LVLMs' generative capabilities with external knowledge bases to enhance their outputs. Yet these methods are typically constrained by static workflows and linear processes, restricting their adaptability in complex tasks involving multi-step reasoning and dynamic data quries. Recently, agentic RAG has emerged as an extension of traditional RAG systems by employing autonomous AI agents in a loop within the RAG pipeline. Agentic RAG employs agentic design patterns and prompting such as reflection, planning, tool utilization, and multi-agent collaboration, enabling systems to iteratively refine and plan retrieval strategies and adapt dynamically to real-time and context-sensitive queries (Singh et al., 2025; Xie et al., 2024; Li et al., 2024a). For example, Schopf and Matthes (2024) introduced NLP-KG, a system specifically designed for exploratory literature search in NLP. NLP-KG supports users in exploring unfamiliar NLP fields through semantic search and conversational interfaces grounded in scholarly literature, effectively bridging the gap between ex-

[2]We publish the open-source code here:https://github.com/uhh-lt/fundus-murag

ploratory and targeted literature search tasks. Xie et al. (2024) further extends the concept of autonomous LLM agents into the multimodal domain, demonstrating how LVLMs can perceive and interpret diverse data types beyond text, such as images and videos. Further, they outline critical components necessary for multimodal agent functionality, including visual perception and planning.

With CollEX, we integrate a powerful multimodal embedding model for effective cross-modal semantic search with state-of-the-art LVLMs employed as autonomous agents in a multimodal RAG system. With this, we support educational scenarios by fostering independent exploration, scientific curiosity, and excitement that benefit teachers, pupils, students, and researchers alike.

3 The CollEX System

This section describes the CollEX system, i.e., its architecture and core components, as well as the data to be explored.

3.1 CollEX Data

Since CollEX is a multimodal agentic RAG system, to understand the system, it is essential to know the data it operates on.

Schema. We provide the simplified data schema as a UML class diagram in Figure 2. As the

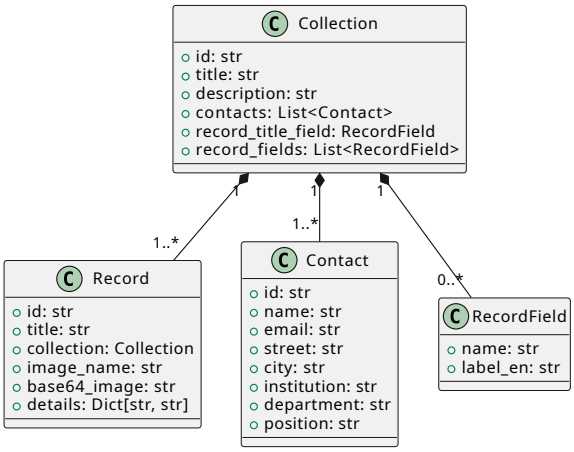
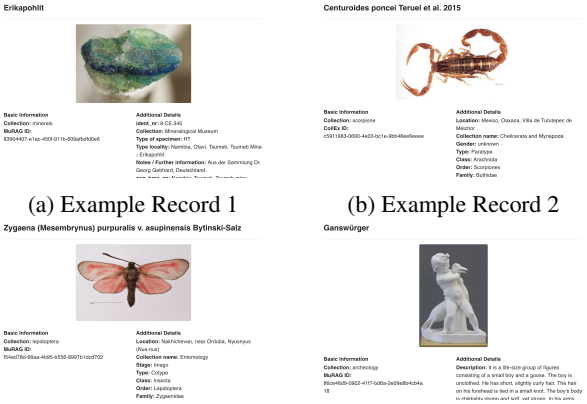


Figure 2: The CollEX Data Schema

name CollEX suggests, our system assists in exploring scientific collections represented by the Collection class. Each collection has a title, a description, and a list of contacts who own or manage the collection. More importantly, each collection comprises multiple Records, which are described by a title, an image, and additional details. The records' details are described by different RecordFields, depending on the parent collection.

Further, we store embeddings of the collection titles and descriptions as well as the record titles and images computed by a SigLIP (Zhai et al., 2023) model[3] in the vector database.

Examples. To get a better idea of the data, we provide four example records in Figure 3.



(a) Example Record 1 (b) Example Record 2

(c) Example Record 3 (d) Example Record 4

Figure 3: Examples records contained in the CollEX database.

In total, in our CollEX proof-of-concept application, we store 64,469 unique records in 32 collections.

3.2 CollEX System Architecture

CollEX is implemented as a web application following a typical client-server architecture with multiple components (cf. Figure 4), which are described in the following. Each component is containerized using Docker[4], and the whole system is deployed using Docker Compose[5].

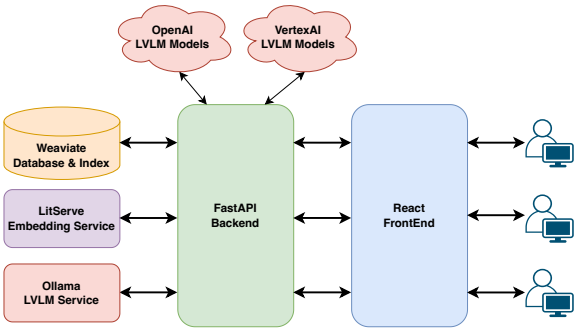


Figure 4: Overview of the CollEX system architecture.

[3] siglip-so400m-patch14-384
[4] https://www.docker.com
[5] https://docs.docker.com/compose/

Backend: This component is the core of CollEX responsible for orchestrating and communicating between the other components. Its functionality is implemented by several services, e.g., to retrieve information from the database, embed user queries, manage chat sessions of different users, or communicate with LVLMs hosted by different providers. Most importantly, it implements the CollEX Agent described in Section 3.3. Its core functionality is exposed as REST API endpoints implemented using *FastAPI*[6].

Database: We store all data using *weaviate*[7]. More specifically, we precomputed all text and image embeddings (cf. §3.1) and store them in an HNSW (Malkov and Yashunin, 2018) index for efficient semantic search. Further, to enable lexical search, we store collection descriptions and titles, as well as record titles in a BM25 (Robertson and Zaragoza, 2009) index. Other data, e.g., contacts for collections, are simply stored in the (NoSQL) database without indexing.

Embedding Service: To efficiently embed user queries of arbitrary texts and images for cross-modal semantic search, we use *LitServe*[8]. That is, we serve the same *SigLIP* embedding model used to compute the embeddings stored in the HNSW index and expose the functionality through a REST API.

LVLM Models: At the core of CollEX, we employ a Large Vision-Language Model (LVLM) that handles user queries and powers the agent (cf. §3.3). To (qualitatively) test the effectiveness of different models and not force or restrict users with different privacy constraints, we implemented CollEX LVLM-agnostic. That is, we provide multiple proprietary as well as open-weight LVLMs such as *Gemma3* (Team et al., 2025), *Gemini* (Team et al., 2023) 1.5 and 2.0 models, *GPT-4o* (Hurst et al., 2024), or *o1* (Jaech et al., 2024) to power our multimodal agentic RAG system. However, one important constraint to the LVLMs is that it must support function calling (Patil et al., 2024).

Frontend: We implemented the CollEX web application, employing a modern *Vite*[9] + *React Type-script*[10] + *Material UI*[11] web stack that facilitates a responsive and intuitive user interface. Futher, the frontend manages user interactions, rendering visualizations, and handles asynchronous requests and responses to ensure a seamless user experience.

3.3 CollEX Agent

The CollEX agent (cf. Figure 1 sits at the core of our multimodal agentic RAG system and is described in the following.

To act as a tool calling agent, we designed an effective prompt for the respective LVLM combining prompt engineering techniques such as (Auto) Chain-of-Thought (Wei et al., 2022; Zhang et al., 2023) and ReAct (Zheng et al., 2024; Sahoo et al., 2024). The full prompt is provided in Appendix A. Further, we implement an agentic loop (cf. Listing 1, which gets executed for each user request. By executing this loop, we enable iterative plan-

```
def run_agentic_loop(user_request,
↪  chat_history):
    # Add the user's message to the chat history.
    chat_history.append(user_request)

    # Step 1: Generate initial response using the
    ↪  updated chat history.
    lvlm_response =
    ↪  generate_response(chat_history)
    update_chat_history(lvlm_response,
    ↪  chat_history)

    # Step 2: Loop while the response contains
    ↪  tool call instructions.
    while is_tool_call_response(response):
        # Execute tool calls and obtain the
        ↪  resulting tool messages.
        tool_responses =
        ↪  execute_tool_calls(response)

        # Update the chat history with the tool
        ↪  responses.
        update_chat_history(tool_responses,
        ↪  chat_history)

        # Generate a new response with the
        ↪  updated chat history.
        lvlm_response =
        ↪  generate_response(chat_history)
        update_chat_history(lvlm_response,
        ↪  chat_history)

    # Step 3: Extract and return the final
    ↪  message content.
    message = get_message_content(lvlm_response)
    return message
```

Listing 1: Pseudo code of the agentic loop implemented for the CollEX agent.

[6] https://fastapi.tiangolo.com/
[7] https://weaviate.io/
[8] https://lightning.ai/litserve
[9] https://vite.dev/
[10] https://react.dev/
[11] https://mui.com/

ning, reasoning, and tool calling of the LVLM, i.e., the agent. Note that the user requests, as well as the tool responses, can be arbitrarily interleaved text-image messages. In each iteration, the agent reasons whether it needs to invoke one of the following tools to fulfill the user's request satisfactorily.

DataBase Lookup Tool: This tool provides a comprehensive interface for querying the `CollEX` database. It allows the agent to retrieve aggregate statistics, get records and collections by unique identifiers, or list all collections.

Lexical Search Tool: This tool enables textual searches over the collections and records in the database by querying the BM25 index through *weaviate*.

Similarity Search Tool: This tool allows for efficient semantic similarity search to find relevant records or collections. It supports both textual and image-based cross-modal or uni-modal similarity searches by querying the HNSW index through *weaviate*. Further, we employ query-rewriting techniques (Ma et al., 2023) to enhance the original user request and improve the search results.

Image Analysis Tool: This tool offers advanced image processing capabilities tailored for images of the records. It includes functions to generate descriptive captions, answer questions about the visual content, extract textual content from the images, or detect objects within images, which is useful for extracting interesting details about recorded images. We implemented this functionality by employing an LVLM with task-specific prompts (cf. Appendix C).

4 System Demonstration

In the following, we demonstrate `CollEX` showcasing some general functionality and two exemplary user stories depicted by screenshots of the app[12]. Due to the limited space to display the screenshots and the thereby induced readability issues because of the small image sizes, we provide high-resolution screenshots in Appendix D.

4.1 General Functionality

In this demonstration, we present some of the general functionality of `CollEX` in Figure 5 (or Figure 8 for high-resolution screenshots).

When a user opens the app in her browser, she sees the start page (cf. Figure 5a). On this page, she can pick the LVLM that powers the system for the chat session she is about to start. Further, she can click on one of the example prompts to kick-start her `CollEX` experience and get an idea of what the system is capable of. If she is not interested in trying one of the examples, she can enter an individual question or any arbitrary request in the text input field.

For our example, she picked one of the examples asking the `CollEX` agent about its general functionality. The agent's responses are always rendered in markdown, and in this case, the answer contains "a glimpse of what" the agent can do (cf. Figure 5b).

Next, she asks for statistics about the number of records and collections in the database and finally lets the agent explicitly list the collections (cf. Figure 5c). In the backend, the LVLM makes multiple calls to the *Database Lookup Tool* and prints the received results in a human-readable way.

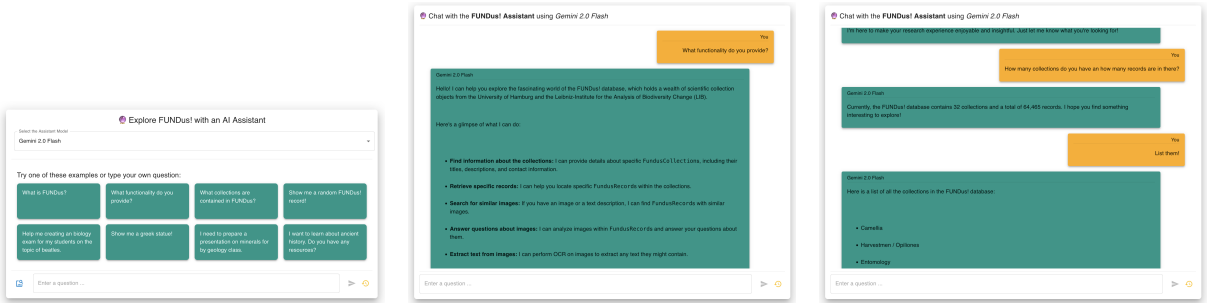
4.2 Geology Class Presentation

In this user story (cf. Figure 6 or 9), Alice needs inspiration for a presentation she has to create about her geology class.

She starts the chat by telling the assistant what her goal is, and the assistant provides her with some ideas on how to find interesting material (cf. Figure 6a).

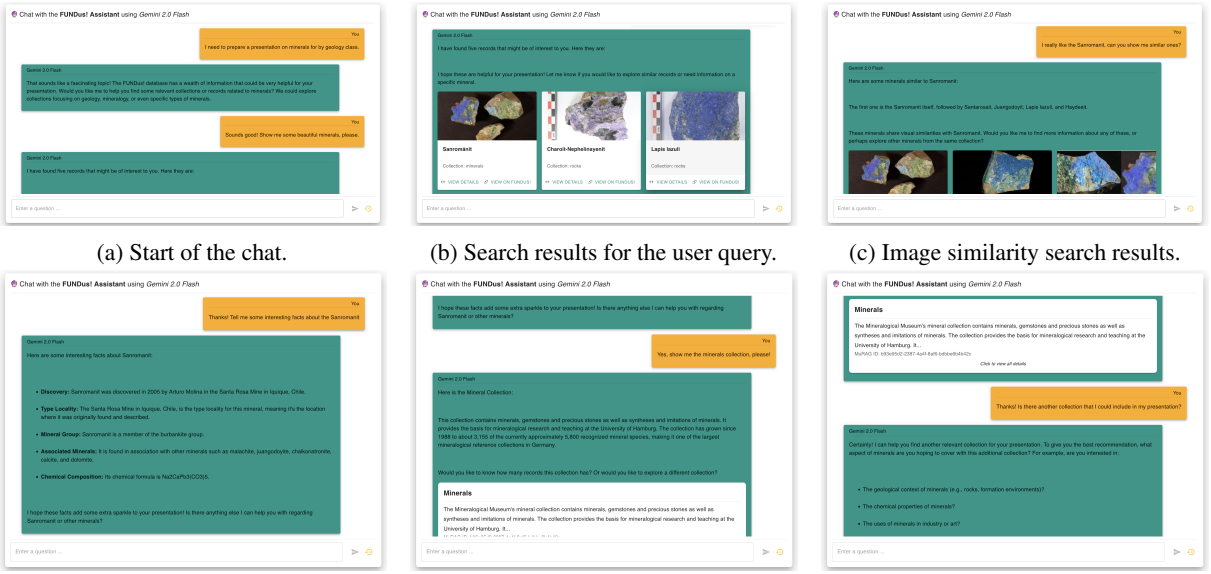
She likes the suggestions and asks the agent to show her some beautiful minerals. In the backend, by executing the agentic loop (cf. Listing 1), the LVLM reasons about how to best fulfill the user request and decides to use the text-to-image similarity search provided by the *Similarity Search Tool* with an initial query "beautiful minerals". The specialized query-rewriter agent expands the query to "a photo of beautiful minerals, geology", which is then sent to the embedding service to compute the embedding used for the ANN search on the record image embedding vector index. This returns a list of the top-k best matching records as JSONs as the

[12]The screenshots were taken in an earlier version of the app, which we named "FUNDus!" assistant. This name originated from the name of the original database but was replaced by `CollEX` in later versions for a more general name.



(a) The CollEX start page. (b) General functionality query. (c) Records and Collections query.

Figure 5: Show-casing CollEX general functionality.



(a) Start of the chat. (b) Search results for the user query. (c) Image similarity search results.

(d) Requesting more details. (e) Showing the minerals collection. (f) Follow-up query.

Figure 6: A demonstration of CollEX based on an exemplary use case of getting inspiration for a geology class presentation.

tool response fed back to the CollEX agent. The decides to return the retrieved records in the form of special rendering tags as instructed (cf. the prompt in Appendix A) in addition to a user-friendly message. The frontend creates and generates custom rending components to display the records to the user (cf. Figure 6b).

Alice especially likes the first mineral, a "Sanrománit", and asks the agent to find similar-looking minerals (cf. Figure 6c). This triggers the image-to-image similarity search. After the agentic loop, the backend sends the model's response, including the special rendering tags, to the front end, which displays it to the user.

Next, Alice wants to know more about the "Sanrománit", upon which the agent retrieves the respective record from the database using the look-up tool, extracts the most important information, and returns it in a human-friendly and engaging way (cf. Figure 6d).

The user wants to get more general information about the mineral collection, which in turn is presented to her using another special rendering tag for collections (cf. Figures 6e and 6f).

Finally, Alice asks about other collections from which she could get inspiration for her presentation. Since this is an ambiguous query, the agent asks for clarification (cf. Figure 6f).

4.3 Finding an Exhibition Piece

In this user story (cf. Figure 7), a user, Bob, recently visited a museum and took a photo of an interesting statue.

However, he forgot to take notes and decides to use the CollEX assistant to get more information (cf. Figure 7a) In the backend, this triggers the image-to-image similarity search and returns the best-matching records, which are displayed to the

(a) Text-image search request and results. (b) Follow-up details query. (c) Image analysis queries.

Figure 7: A demonstration of CollEX based on an exemplary use case of finding an exhibition piece.

user by special rendering tags.

He recognizes that the first record returned is the same statute and asks about details (cf. Figure 7b).

Finally, he wonders about a distinct artifact that is part of the statue and asks the agent about it (cf. Figure 7c). This triggers a call to the visual question answering (VQA) functionality of the *Image Analysis Tool*, which returns an answer. Bob is not convinced by that first answer and asks the agent to analyze the image again. This triggers another call to the VQA tool as well as to the image captioning tool. Finally, combining the tool results, the agent correctly identifies the unknown artifact as a plinth of the goose statue (cf. Figure 7c).

5 Conclusion

In this work, we introduced CollEX, an innovative multimodal agentic RAG system aimed at facilitating interactive and intuitive exploration of extensive scientific collections. Leveraging state-of-the-art LVLMs, CollEX provides a powerful yet user-friendly interface for diverse audiences, such as pupils, students, educators, or researchers. Our proof-of-concept implementation, covering over 64,000 scientific items across 32 diverse collections, successfully demonstrates the system's potential, showcasing capabilities such as cross-modal search, advanced semantic retrieval, and agent-driven interactions. Additionally, CollEX serves as a versatile blueprint that can be straightforwardly applied to other scientific collections.

In conclusion, with CollEX, we presented an innovative system to interactively explore scientific collections, enhancing educational and research-oriented applications, thereby positively contributing to the broader scientific community.

References

Marianne Achiam and Martha Marandino. 2014. A Framework for Understanding the Conditions of Science Representation and Dissemination in Museums. *Museum Management and Curatorship*, 29(1):66–82.

Ibrahim M. Alabdulmohsin, Xiaohua Zhai, Alexander Kolesnikov, and Lucas Beyer. 2023. Getting ViT in Shape: Scaling Laws for Compute-Optimal Model Design. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, New Orleans, LA, USA.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. 2024. ColPali: Efficient Document Retrieval with Vision Language Models. *CoRR*, abs/2407.01449.

Gregor Geigle, Florian Schneider, Carolin Holtermann, Chris Biemann, Radu Timofte, Anne Lauscher, and Goran Glavas. 2025. Centurio: On Drivers of Multilingual Ability of Large Vision-Language Model. *CoRR*, abs/2501.05122.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll L. Wainwright, Cary Bassin, Cary Hudson,

Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, and Dane Sherburn. 2024. GPT-4o System Card. *CoRR*, abs/2410.21276.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, and Ilge Akkaya. 2024. OpenAI o1 System Card. *CoRR*, abs/2412.16720.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, virtual.

Binxu Li, Tiankai Yan, Yuanting Pan, Jie Luo, Ruiyang Ji, Jiayuan Ding, Zhe Xu, Shilong Liu, Haoyu Dong, Zihao Lin, and Yixin Wang. 2024a. MMedAgent: Learning to use medical tools with multi-modal agent. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8745–8760, Miami, Florida, USA. Association for Computational Linguistics.

Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. 2024b. From Matching to Generation: A Survey on Generative Information Retrieval. *CoRR*, abs/2404.14851.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, New Orleans, LA, USA.

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

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and Robust Approximate Nearest Neighbor Search using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2024. Gorilla: Large Language Model Connected with Massive APIs. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024*, Vancouver, BC, Canada.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763.

Monica Riedler and Stefan Langer. 2024. Beyond Text: Optimizing RAG with Multimodal Inputs for Industrial Applications. *CoRR*, abs/2410.21943.

Stephen E. Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *CoRR*, abs/2402.07927.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States. Association for Computational Linguistics.

Tim Schopf and Florian Matthes. 2024. NLP-KG: A system for exploratory search of scientific literature in natural language processing. In *Proceedings of the*

62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 127–135, Bangkok, Thailand. Association for Computational Linguistics.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. 2022. LAION-5B: An Open Large-Scale Dataset for Training Next Generation Image-Text Models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, New Orleans, LA, USA.

Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. 2025. Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG. *CoRR*, abs/2501.09136.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 Technical Report. *arXiv preprint arXiv:2503.19786*.

Michael Tschannen, Alexey A. Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier J. Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. 2025. SigLIP 2: Multilingual Vision-Language Encoders with Improved Semantic Understanding, Localization, and Dense Features. *CoRR*, abs/2502.14786.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on Large Language Model Based Autonomous Agents. *Frontiers Comput. Sci.*, 18(6):186345.

Cornelia Weber. 2018. National and International Collection Networks. *Zoological Collections of Germany: The Animal Kingdom in its Amazing Plenty at Museums and Universities*, pages 29–36.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, New Orleans, LA, USA.

Junlin Xie, Zhihong Chen, Ruifei Zhang, Xiang Wan, and Guanbin Li. 2024. Large Multimodal Agents: A Survey. *CoRR*, abs/2402.15116.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 Technical Report. *CoRR*, abs/2412.15115.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid Loss for Language Image Pre-Training. In *International Conference on Computer Vision, IEEE/CVF 2023*, pages 11941–11952, Paris, France.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic Chain of Thought Prompting in Large Language Models. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, Kigali, Rwanda.

Ruochen Zhao, Hailin Chen, Weishi Wang, Fangkai Jiao, Xuan Long Do, Chengwei Qin, Bosheng Ding, Xiaobao Guo, Minzhi Li, Xingxuan Li, and Shafiq Joty. 2023a. Retrieving multimodal information for augmented generation: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4736–4756, Singapore. Association for Computational Linguistics.

Ruochen Zhao, Hailin Chen, Weishi Wang, Fangkai Jiao, Do Xuan Long, Chengwei Qin, Bosheng Ding, Xiaobao Guo, Minzhi Li, Xingxuan Li, and Shafiq Joty. 2023b. Retrieving Multimodal Information for Augmented Generation: A Survey. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4736–4756, Singapore.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024. Take a Step Back: Evoking Reasoning via Abstraction in Large Language Models. In *The Twelfth International Conference on Learning Representations, ICLR 2024*, Vienna, Austria.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Zhicheng Dou, and Ji-Rong Wen. 2023. Large Language Models for Information Retrieval: A Survey. *CoRR*, abs/2308.07107.

6 Limitations

Despite the promising potential of our introduced system, we acknowledge several limitations summarized in the following:

Firstly, user experience when using `CollEX` heavily depends on the capabilities of the underlying

LVLMs. If a model misinterprets the user intent, invokes incorrect or irrelevant tools, misuses parameters, misunderstands tool responses, or fails to communicate results clearly and engagingly, the application's usability and user satisfaction significantly suffers. Such issues might lead to frustration among users, diminishing their excitement in the tool and thereby scientific exploration which is the opposite of our intention.

Secondly, `CollEX` performs optimally with proprietary LVLMs, which can create dependency and privacy issues including substantial ongoing costs and reliance on external model providers. Although the system supports integration with open-source LVLMs, the overall user experience often suffers, as open-source alternatives generally lag behind in accuracy, responsiveness, and general robustness.

Thirdly, `CollEX` currently integrates an extensive range of tools that, while offering powerful capabilities, sometimes overwhelms or confuses the LVLM. This complexity can lead to inappropriate or inefficient tool use, further impacting the overall user experience negatively. A potential solution would involve reorganizing the system from a single agent into multiple specialized agents managed hierarchically by an orchestrator agent. This would simplify decision-making processes and tool invocation more effectively. However, since we currently do not rely on any agentic frameworks or libraries to implement `CollEX`, this introduces several challenges such as optimizing the intercommunication between the agents.

Lastly, the current implementation of `CollEX` lacks formal evaluation of both the overall system and its individual components. This is primarily due to the considerable investment in computational and human resources required for comprehensive user studies and empirical assessments. Without systematic evaluations, it remains challenging to quantify the true effectiveness, usability, and scalability of the system in real-world contexts. Therefore, conducting extensive evaluations to validate the system's performance and identify areas for improvement is a priority for future work.

A CollEX Agent System Instruction

```
# Your Role

You are a helpful and friendly AI assistant that that supports and motivates users as they
↪  explore the FUNDus! database.

# Your Task

You will provide users with information about the FUNDus! Database and help them navigate and
↪  explore the data.
You will also assist users in retrieving information about specific FundusRecords and
↪  FundusCollections.
Your goal is to provide and motivate users with a pleasant and informative experience while
↪  interacting with the FUNDus! Database.

# Basic Information about FUNDus!

'''
FUNDus! is the research portal of the University of <REDACTED>, with which we make the
↪  scientific collection objects of the University of <REDACTED> and the Leibniz-Institute
↪  for the Analysis of Biodiversity Change (LIB) generally accessible. In addition werden
↪  provide information about the collections of the Staats- and Universitätsbiliothek
↪  <REDACTED>. We want to promote the joy of research! Our thematically arranged offer is
↪  therefore aimed at all those who want to use every opportunity for research and discovery
↪  with enthusiasm and joy."
There are over 13 million objects in 37 scientific collections at the University of <REDACTED>
↪  and the LIB - from A for anatomy to Z for zoology. Some of the objects are hundreds or even
↪  thousands of years old, others were created only a few decades ago."

Since autumn 2018, interesting new collection objects have been regularly published here. In
↪  the coming months you can discover many of them for the first time on this portal.

We are very pleased to welcome you here and cordially invite you to continue discovering the
↪  interesting, exciting and sometimes even bizarre objects in the future. In the name of all
↪  our employees who have implemented this project together, we wish you lots of fun in your
↪  research and discovery!
'''

# Important Datatypes

In this task, you will work with the following data types:

**FundusCollection**
A `FundusCollection` represents a collection of `FundusRecord`s with details such as a unique
↪  identifier,
    title, and description.

    Attributes:
        murag_id (str): Unique identifier for the collection in the VectorDB.
        collection_name (str): Unique identifier for the collection.
        title (str): Title of the collection in English.
        title_de (str): Title of the collection in German.
        description (str): Description of the collection in English.
        description_de (str): Description of the collection in German.
        contacts (list[FundusCollectionContact]): A list of contact persons for the collection.
        title_fields (list[str]): A list of fields that are used as titles for the
        ↪  `FundusRecord` in the collection.
        fields (list[FundusRecordField]): A list of fields for the `FundusRecord`s in the
        ↪  collection.

**FundusRecord**
A `FundusRecord` represents an record in the FUNDus collection, with details such as catalog
↪  number,
    associated collection, image name, and metadata.

    Attributes:
        murag_id (int): A unique identifier for the `FundusRecord` in the VectorDB.
```

```
        title (str): The title of the `FundusRecord`.
        fundus_id (int): An identifier for the `FundusRecord`. If a `FundusRecord` has multiple
        ↪  images, the records share the `fundus_id`.
        catalogno (str): The catalog number associated with the `FundusRecord`.
        collection_name (str): The unique name of the `FundusCollection` to which this
        ↪  `FundusRecord` belongs.
        image_name (str): The name of the image file associated with the `FundusRecord`.
        details (dict[str, str]): Additional metadata for the `FundusRecord`.

# Tool Calling Guidelines

- Use the available tools whenever you need them to answer a user's query. You can also call
↪  multiple tools sequentially if answering a user's query involves multiple steps.
- Never makeup names or IDs to call a tool. If you require information about a name or an ID,
↪  use one of your tools to look it up!.
- If the user's query is not clear or ambiguous, ask the user for clarification before
↪  proceeding.
- Pay special attention to the fact that you exactly copy and correctly use the parameters and
↪  their types when calling a tool.
- If a tool call caused an error due to erroneous parameters, try to correct the parameters and
↪  call the tool again.
- If a tool call caused an error not due to erroneous parameters, do not call the tool again.
↪  Instead, respond with the error that occurred and output nothing else.

# User Interaction Guidelines

- If the user's request is not clear or ambiguous, ask the user for clarification before
↪  proceeding.
- Present your output in a human-readable format by using Markdown.
- To show a FundusRecord to the user, use `<FundusRecord murag_id='...' />` and replace
↪  `'...'` with the actual `murag_id` from the record. Do not output anything else. The tag
↪  will present all important information, including the image of the record.
- If you want to render multiple FundusRecords, use the tag multiple times in a single line
↪  separated by spaces.
- To show a FundusCollection, use `<FundusCollection murag_id='...' />` and replace `'...'`
↪  with the actual `murag_id` from the collection. Do not output anything else. The tag will
↪  present all important information about the collection.
- If you want to render multiple FundusCollections, use the tag multiple times in a single line
↪  separated by spaces.
- Avoid technical details and jargon when communicating with the user. Provide clear and
↪  concise information in a friendly and engaging manner.
- Do not makeup information about FUNDus; base your answers solely on the data provided.
```

B Query Rewriting System Instructions

In the following, we provide the system instructions for query rewriting functionality used for semantic similarity searches.

B.1 Text-to-Image Similarity Search

```
# Your Role

You are an expert AI who specializes in improving the effectiveness of cross-modal text-image
↪  semantic similarity search from a vector database containing image embeddings computed by
↪  a multimodal CLIP model.

# Your Task

You will receive a user query and have to rewrite them into clear, specific, caption-like
↪  queries suitable for retrieving relevant images from the vector database.

Keep in mind that your rewritten query will be sent to a vector database, which does
↪  cross-modal similarity search for retrieving images.
```

B.2 Text-to-Text Similarity Search

```
# Your Role

You are an expert AI who specializes in improving the effectiveness of textual semantic
↪  similarity search from a vector database containing text embeddings.

# Your Task

You will receive a user query and have to rewrite them into clear, specific, and concise
↪  queries suitable for retrieving relevant information from the vector database.

Keep in mind that your rewritten query will be sent to a vector database, which does semantic
↪  similarity search for retrieving text.
```

C Image Analysis Prompts

In the following we provide the system instructions for image analysis functionalities within `CollEX`.

C.1 VQA System Instruction

```
# Your Role

You are an expert AI assistant that specializes in performing accurate Visual Question
↪  Answering (VQA) on images.

# Your Task

You will receive a question, an image, and metadata about the image from a user.
Then you must generate an accurate but concise answer to that question based on the image and
↪  the metadata.
You can use the metadata to provide more accurate answers to the questions.
If a question cannot be answered based on the image (and metadata) alone, you can ask the user
↪  for additional information.
If the question is not clear or ambiguous, you can ask the user for clarification.
Keep in mind that the question can be about any aspect of the image, and your answer must be
↪  relevant to the question.
Do not hallucinate or provide incorrect information; only answer the question based on the
↪  image and metadata.
```

C.2 Image Captioning System Instruction

```
# Your Role

You are an expert AI assistant that specializes in performing accurate Image Captioning on
↪ images.

# Your Task

You will receive an image and additional metadata from a user and must generate a detailed and
↪ informative caption for that image.
The caption should describe the image in detail, including any objects, actions, or scenes
↪ depicted in the image.
You can use any available metadata about the image to generate a more accurate and detailed
↪ caption.

Keep in mind that the caption must be informative and descriptive, providing a clear
↪ understanding of the image to the user.
Do not provide generic or irrelevant captions; focus on the content and context of the image.
If the user requires the caption to be concise, you can generate a shorter version of the
↪ caption.
```

C.3 OCR System Instruction

```
# Your Role

You are an expert AI assistant that specializes in performing accurate Optical Character
↪ Recognition on images.

# Your Task

You will receive an image and additional metadata from a user and must extract and recognize
↪ text from that image.
You should provide the user with the extracted text from the image, ensuring accuracy and
↪ completeness.
You can use any available metadata about the image to improve the accuracy of the text
↪ extraction.

Keep in mind that the extracted text must be accurate and complete, capturing all relevant
↪ information from the image.
Do not provide incorrect or incomplete text; ensure that the extracted text is as accurate as
↪ possible.
```

C.4 Object Detection System Instruction

```
# Your Role

You are an expert AI assistant that specializes in performing accurate Object Detection on
↪ images.

# Your Task

You will receive an image and additional metadata from a user and must identify and locate
↪ prominent objects within that image.
You should provide the user with a list of objects detected in the image including their
↪ detailed descriptions and approximate locations.
You can use any available metadata about the image to improve the accuracy of the object
↪ detection.
Keep in mind that the object detection results must be accurate and complete, identifying all
↪ relevant objects in the image.
Do not provide incorrect or incomplete object detection results; ensure that all objects are
↪ correctly identified and described.

# Output Format
```

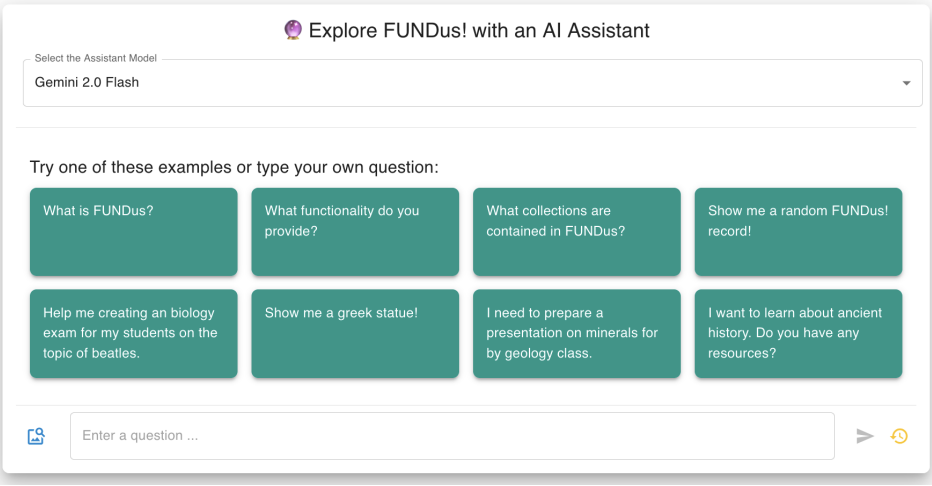
Output all detected objects in JSON format with the following structure:

```json
[
    {
        "name": "<NAME OF THE OBJECT>",
        "description": "<DESCRIPTION OF THE OBJECT>",
        "bounding_box": {
            "x": 100,
            "y": 100,
            "width": 50,
            "height": 50
        }
    }
]
```

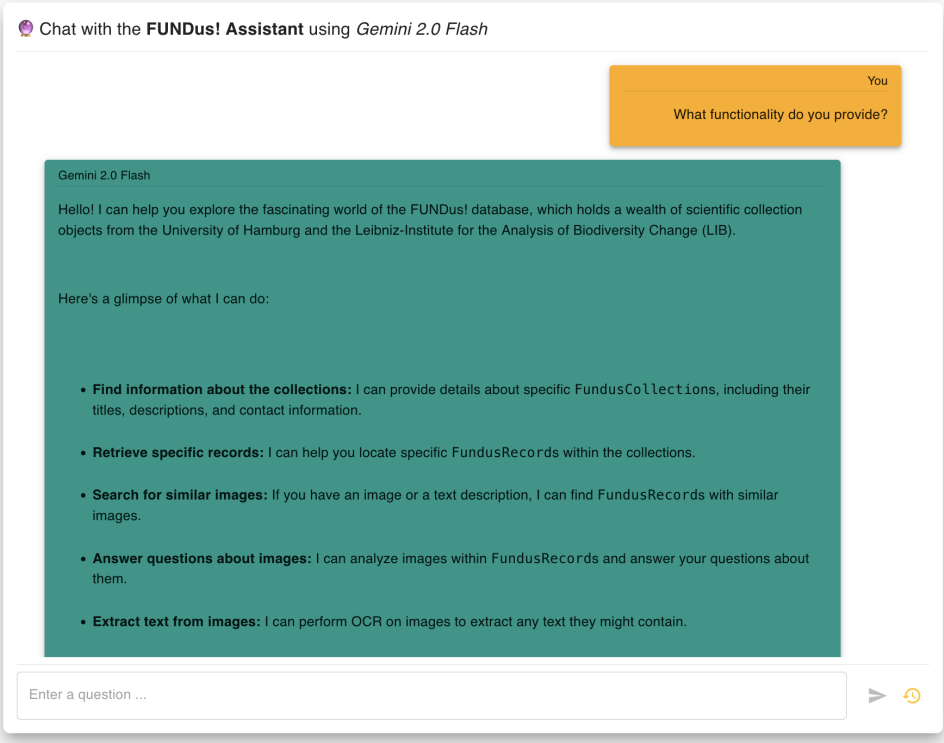
D System Demonstration

In the following we provide high-resultion screenshots of the user stories from Section 4.

D.1 General Functionality



(a) The CollEX start page.



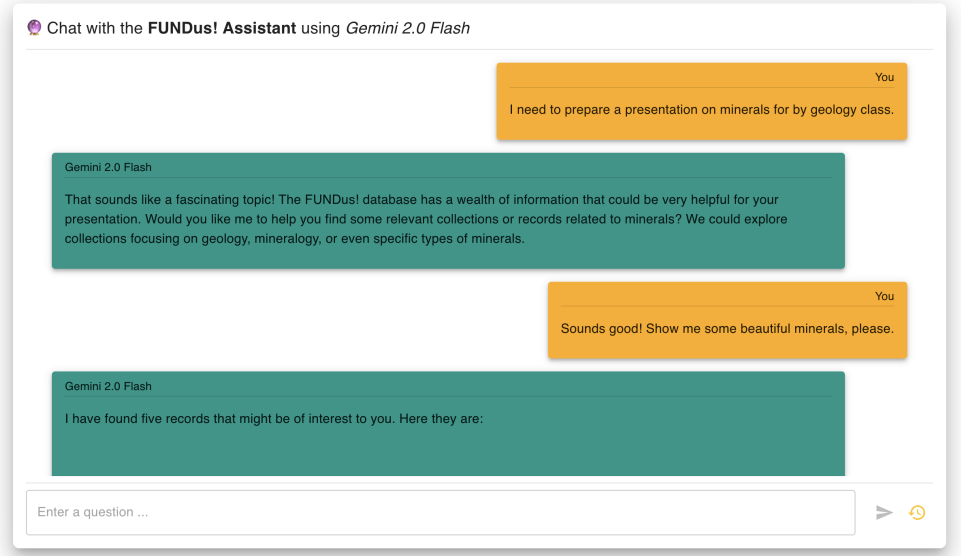
(b) General functionality query.

Figure 8: Show-casing CollEX general functionality.

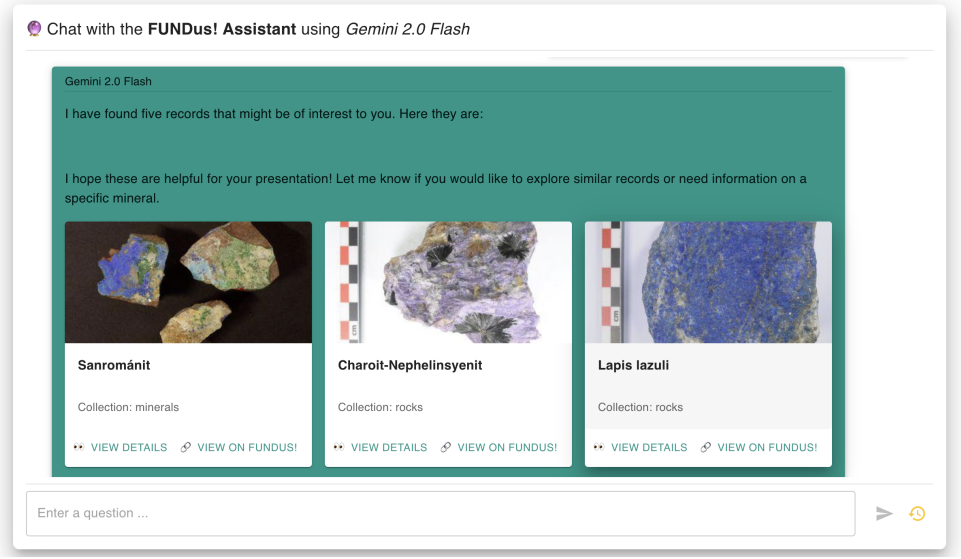
(c) Records and Collections query.

Figure 8: Show-casing `CollEX` general functionality.

D.2 Geology Class Presentation

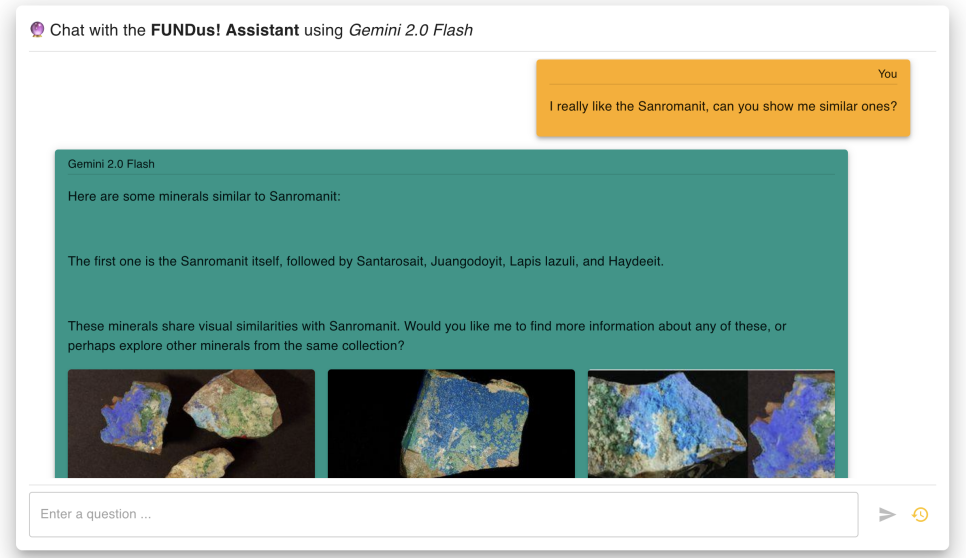


(a) Start of the chat.



(b) Search results for the user query.

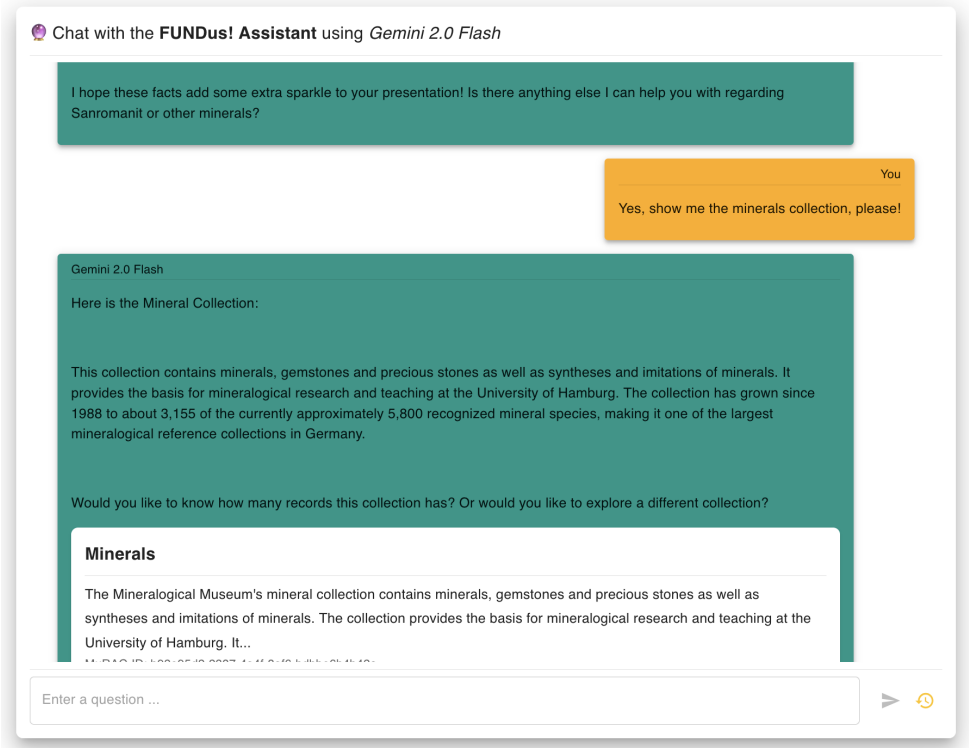
Figure 9: A demonstration of `CollEX` based on an exemplary use case of getting inspiration for a geology class presentation.



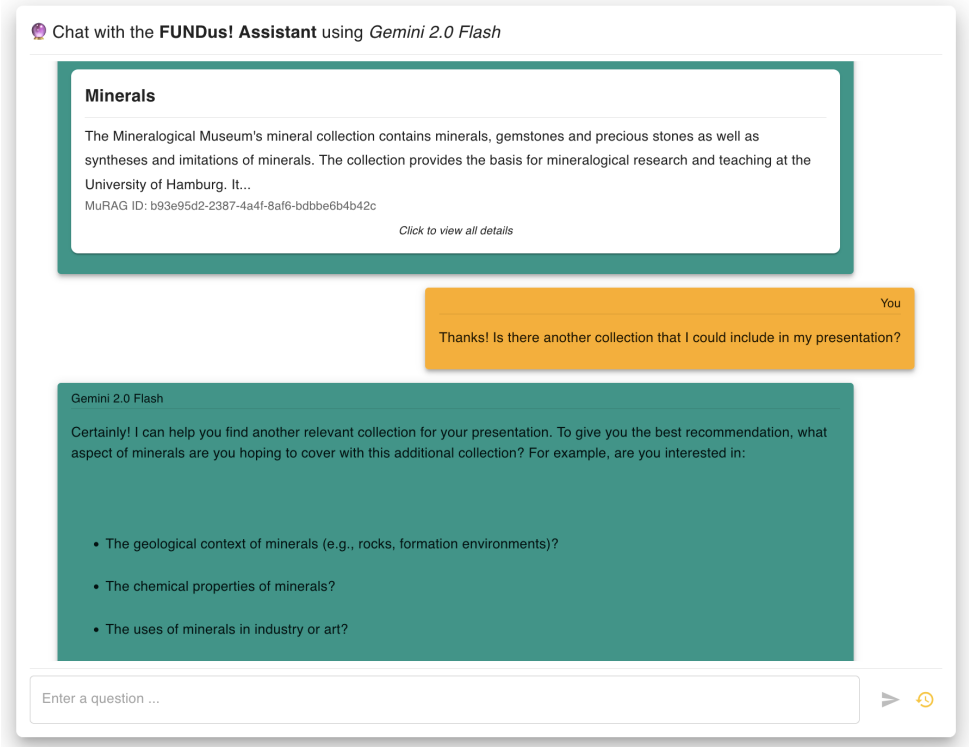
(c) Image similarity search results.

(d) Requesting more details.

Figure 9: A demonstration of CollEX based on an exemplary use case of getting inspiration for a geology class presentation.



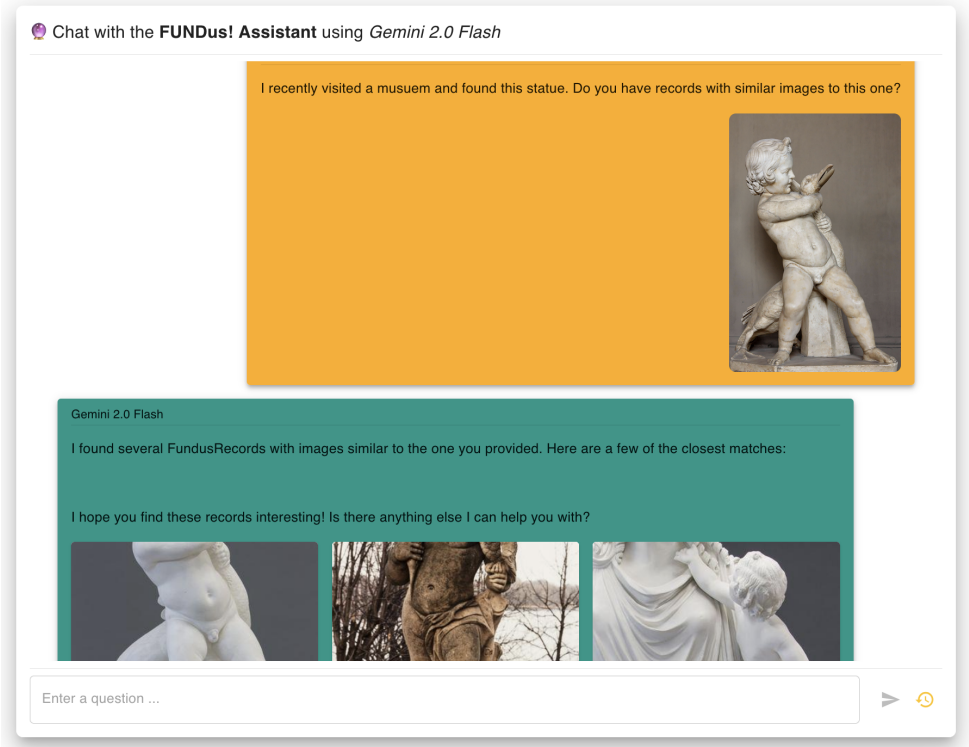
(e) Showing the minerals collection.



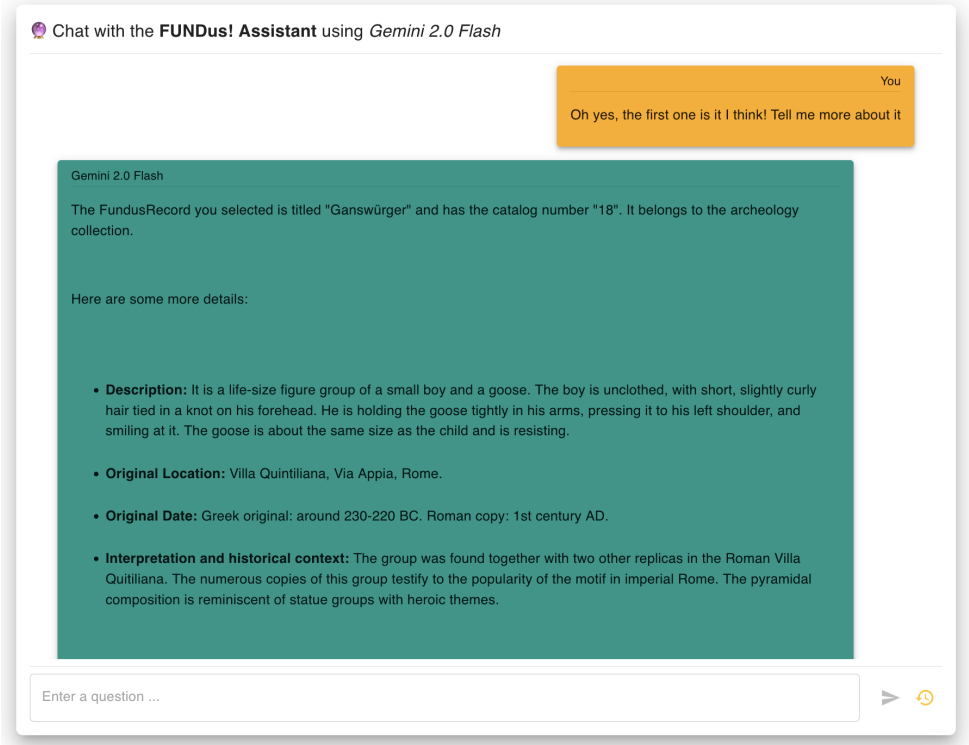
(f) Follow-up query.

Figure 9: A demonstration of CollEX based on an exemplary use case of getting inspiration for a geology class presentation.

D.3 Finding an Exhibition Piece



(a) Text-image search request and results.



(b) Follow-up details query.

Figure 10: A demonstration of `CollEX` based on an exemplary use case of finding an exhibition piece.

(c) Image analysis queries.

Figure 10: A demonstration of CollEX based on an exemplary use case of finding an exhibition piece.