MindTrellis: Co-Creating Knowledge Structures with AI through Interactive Visual Exploration

Xiang Li*
University of Waterloo
Waterloo, Ontario, Canada
x247li@uwaterloo.ca

Cara Li*
University of Waterloo
Waterloo, Ontario, Canada
cy5li@uwaterloo.ca

Emily Kuang
York University
Toronto, Ontario, Canada
ekuang@yorku.ca

Can Liu
Nanyang Technological University
Singapore, Singapore
can.liu@ntu.edu.sg

Jian Zhao
University of Waterloo
Waterloo, Ontario, Canada
jianzhao@uwaterloo.ca

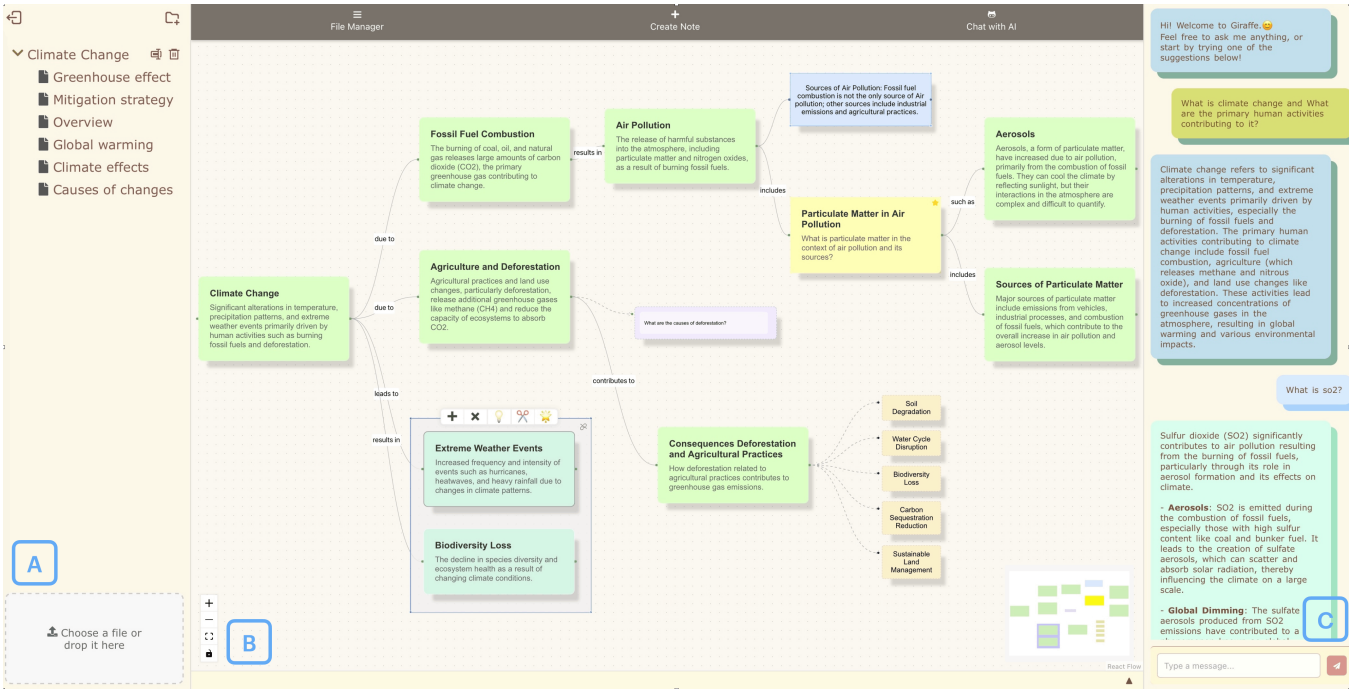


Figure 1: MindTrellis is an interactive visual system that enables human-AI collaborative knowledge construction, where users can both query information and contribute insights to an evolving knowledge graph. (a) File Manager – allows users to upload and organize documents. (b) Knowledge Canvas – displays the knowledge graph where users can explore and directly manipulate nodes and relationships. (c) Chat Panel – enables users to query information, request expansions, and issue modification commands through natural language.

*Both authors contributed equally to this research.

DIS '26, June 13–17, 2026, Singapore, Singapore

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ABSTRACT

Synthesizing information from multiple documents into structured understanding is inherently iterative, yet current approaches provide limited support. LLM-based systems let users query information but produce structures that users cannot reshape; manual tools like mind maps offer full control but lack intelligent assistance; and commercial tools have begun combining retrieval with user contribution, but not within a unified visual knowledge structure. We present MindTrellis, an interactive visual system that addresses this gap by letting users and AI collaboratively build a knowledge graph combining document-derived and user-contributed knowledge. Users can query the graph to retrieve document-grounded

information, and contribute new concepts, relationships, and hierarchical organization to reflect their developing understanding. A multi-agent pipeline coordinates intent disambiguation, knowledge placement, and coherence maintenance across both pathways. In a controlled study where 12 participants created slide decks, MindTrellis outperformed a retrieval-only baseline in knowledge organization and cognitive load, with participants valuing progressive graph expansion and the ability to integrate their own insights.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; *Visualization systems and tools*; • **Computing methodologies** → Natural language processing.

KEYWORDS

Knowledge Exploration; Interactive Visualization; Human-AI Collaboration; Multi-Agent System

ACM Reference Format:

Xiang Li, Cara Li, Emily Kuang, Can Liu, and Jian Zhao. 2026. MindTrellis: Co-Creating Knowledge Structures with AI through Interactive Visual Exploration. In *Designing Interactive Systems Conference (DIS '26), June 13–17, 2026, Singapore, Singapore.* ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

As digital resources continue to expand, knowledge workers face increasing challenges in understanding and synthesizing information from multiple sources. Developing a coherent understanding is inherently an iterative process: users explore information, form mental models, refine their understanding, and reorganize their knowledge as they learn [40]. However, current tools provide limited support for this evolving process, creating a gap between how human understanding develops and how systems facilitate knowledge construction. Traditional visualization tools based on mind maps [4] and concept maps [37] allow users to construct and modify knowledge representations, offering full control over structure. However, they lack sufficient intelligent assistance, largely relying on users' manual operations, which makes the process tedious and cognitively demanding when dealing with complex, multi-source information. In recent years, researchers have developed automatic knowledge graph construction methods that use extraction algorithms to identify entities and relationships from text corpora [18], but this process involves minimal human participation and produces structures optimized for machine consumption rather than human understanding. Since the emergence of Large Language Models (LLMs), retrieval-augmented generation (RAG) systems [27] have enabled users to query knowledge bases through natural language. Recent systems such as Graphologue [21], Sensecape [51], Selenite [29], and Luminate [50] further integrate LLMs with graphical representations to support visual exploration of complex information. Yet these systems treat the underlying knowledge structure as *largely static*: users can retrieve information and receive AI-generated visualizations, but cannot add their own concepts, modify relationships, or reorganize how the structure is arranged. The broader pattern of users shaping the same representations they consume from is well established in interactive systems [17, 59], and commercial tools such as NotebookLM and Notion AI already let users both query and contribute to underlying knowledge stores. These commercial tools, however, distribute retrieval and contribution across separate panes or database views rather than integrating them within a single visual knowledge structure.

Multi-document knowledge construction is fundamentally iterative [42]: users externalize their developing mental models [23], test relationships between concepts [5, 37], and continuously reorganize their representations as understanding deepens [3, 40]. Research on knowledge visualization suggests that spatial representations reduce cognitive effort [25] and that the cognitive value of working with multiple representations lies in the transformations between them [3]. A system designed to support this process should therefore integrate AI-assisted retrieval and user contribution within a shared visual structure, so that users can see how new content relates to existing knowledge and reorganize the whole as their understanding evolves.

We present *MindTrellis*, an interactive visual system for multi-document knowledge construction whose design centers on a visual knowledge graph as the primary shared artifact (Fig. 1). In one direction, users can *query* the knowledge base to retrieve information grounded in their uploaded documents. In the other direction, users can *contribute* to the knowledge structure by adding new concepts, modifying relationships, and reorganizing the hierarchy to reflect their developing understanding; the system simultaneously synchronizes these contributions to the underlying knowledge base to maintain consistency between the visual representation and stored knowledge. The resulting representation is what we call a *co-created knowledge graph*: a hybrid of document-derived content and user-contributed insights that evolves through the collaboration between human understanding and AI assistance.

However, enabling this co-creation introduces new technical challenges. User input is inherently ambiguous: a query like *"Expand the effects of deforestation on causing global warming"* could be a request for information retrieval, a command to expand the knowledge structure, or both simultaneously. When users contribute new knowledge, the system needs to determine where this information belongs within the existing taxonomy, which requires understanding both the new content and the current graph topology. As the knowledge graph evolves through user contributions, the system must also maintain coherence and consistency, avoiding contradictions and redundancies. To address these challenges, we developed a *multi-agent pipeline* in MindTrellis where specialized components collaborate to support seamless bidirectional interaction: classifying user intent, routing inputs to appropriate processing pipelines, handling retrieval at multiple granularity levels, and executing structural modifications while maintaining graph coherence.

Based on the co-created knowledge graph and the multi-agent system, MindTrellis supports two complementary interactions for controlling and modifying the knowledge graph, within a unified environment. Through the chat interface, users can issue natural language commands to query information, add nodes, or restructure relationships. Through the visual canvas, users can directly manipulate [19] the graph by clicking to select nodes, dragging to reposition elements, and using toolbar controls to expand topics or create connections.

To evaluate MindTrellis, we conducted a controlled user study with 12 participants comparing our system against a baseline system that combines a RAG interface with automatically generated graph visualizations. The baseline supports the query direction, where users can ask questions and view document-grounded answers as node diagrams, but does not allow users to modify the generated structure or add their own concepts. Participants rated MindTrellis significantly higher on knowledge organization effectiveness and reported lower frustration when exploring unfamiliar topics. Qualitative feedback highlighted that progressive expansion of the knowledge graph, where new nodes branch from existing ones rather than appearing all at once, reduced participants' cognitive overload and helped maintain a coherent mental model. Participants also valued the ability to integrate their own insights into the evolving structure. These findings suggest that enabling users to actively shape knowledge representations, rather than passively consuming system-generated structures, can meaningfully improve outcomes in information exploration tasks.

In summary, our contributions in this paper include:

- The design and implementation of MindTrellis, an interactive system for knowledge construction from multi-document sources, featuring a visual knowledge graph where document-derived and user-contributed knowledge coexist, manipulable through both natural language and direct manipulation.
- A multi-agent pipeline that addresses the technical challenges of intent parsing, knowledge placement, and coherence maintenance, validated through quantitative evaluation.
- Findings from a controlled user study demonstrating that MindTrellis improves knowledge organization effectiveness and reduces cognitive load compared to retrieval-only baselines.

2 RELATED WORK

2.1 Knowledge Visualization and Externalization

As both digital and physical resources continue to expand, people increasingly recognize the value of visualizing knowledge [14, 32] to aid in learning and understanding complex information [5, 43]. Knowledge visualization provides a structured way to externalize thoughts, which is particularly useful for understanding complex or multi-faceted subjects [54]. Different forms of visualization, especially graph-based representations, can significantly enhance critical thinking [22, 30] and comprehension [11, 41], which includes concept maps [9, 26, 38] and mind maps [15, 56]. Overall, these visualization techniques support memory retention and comprehension as well as facilitate deeper cognitive engagement and a more structured understanding of complex topics. However, not all visual forms are equally effective, and the choice of representation can shape what people notice, how they reason, and what knowledge they construct.

Carneiro et al. [6] compared textual and graphical representations for argument analysis and found that graph-based interfaces better support reasoning over non-linear argument structures, where relationships between claims are difficult to convey through sequential text. Binks et al. [3] studied how users move between map-based and textual representations during essay writing, finding that the cognitive value lies not within any single representation but in the *representational transformations* between them—the process of re-expressing and reorganizing knowledge across formats. Their characterization of transformation properties (cardinality, explicitness, and representation type change) suggests that systems for knowledge work should facilitate fluid movement between complementary representations. Larkin and Simon [25] offered an earlier theoretical account of these effects, showing that spatial and diagrammatic representations can reduce cognitive effort by supporting perceptual inference and reducing search compared to informationally equivalent text. The above work motivates us to design MindTrellis that goes beyond static visualization to support interactive construction and manipulation of knowledge structures.

2.2 Bidirectional Interaction Between Representations

Bidirectional interaction between coupled representations is a well-established design pattern in interactive systems and programming languages [16]. In such systems, users can shape the same artifact they consume from, and the system keeps both views synchronized. Sketch-n-Sketch [17] enables users to write programs that generate SVG graphics and then directly manipulate the rendered output; the system infers corresponding program updates through trace-based synthesis, keeping code and visual artifact synchronized. Penrose [59] maps mathematical notation to diagrams through a trio of domain-specific languages and constraint-based optimization, translating formal specifications into visual layouts. B2 [58] bridges code and interactive visualizations in computational notebooks by treating data queries as a shared intermediate representation, so that interactions with a chart reify as code and vice versa. The pattern extends to other domains as well: Cascaval et al. [7] apply bidirectional editing to parametric CAD programs, where users can manipulate geometry directly and the system solves an inverse problem to update program parameters. In each of these systems, two views of a single underlying artifact are kept in sync through deterministic or constraint-based mechanisms—a pattern we refer to as *representational synchronization*. The mapping is mechanical rather than interpretive, the system does not need to infer what the user intends, because the editing modality is spatially explicit (a code panel versus an output canvas) and the artifact on both sides is the same.

As bidirectional interaction extends from structured authoring and design tools into constructing knowledge structures from multiple documents, the nature of the pattern changes. The coupled representations are no longer two views of one artifact but two distinct kinds of content: source documents and an evolving knowledge graph. Users contribute concepts that may not appear in any source document [40], so the graph becomes a hybrid of document-derived and user-contributed knowledge—a structure richer than either source alone. Because retrieval and contribution share the same natural language input channel, the system must determine which activity a given input represents. A statement like *"Expand the effects of deforestation on the cause global warming"* could be a question seeking retrieval or a suggestion to add a new concept to the graph. Shahriari et al. [47] studied natural language interaction for editing visual knowledge graphs and found that distinguishing between edit intent and information queries is a central design

challenge. The transformation between representations is therefore AI-mediated and semantically interpretive, requiring the system to classify user intent before determining how to act. Intent disambiguation, coherent placement within an evolving structure, and maintenance of a hybrid artifact are requirements specific to knowledge-level bidirectionality; they do not arise in the representational synchronization paradigm, where editing modalities are structurally partitioned and mappings are deterministic.

2.3 Interactive Systems for Knowledge Management

Traditional knowledge management tools primarily support static representations, requiring users to manually create, organize, and update all connections and relationships [9]. However, manual construction becomes cognitively demanding as datasets grow larger and more complex, and the resulting structures often lack the visual scaffolding needed to help users organize and make sense of information across sources [14]. A parallel line of research has focused on automatic knowledge graph construction, where systems extract entities and relationships from text corpora using named entity recognition, relation extraction, and knowledge base population techniques [20]. While these approaches can process large volumes of documents efficiently, they are designed primarily for machine consumption—producing structured databases optimized for computational queries—rather than human knowledge building. The resulting knowledge graphs reflect algorithmic decisions with no mechanism for incorporating human insight or adapting to individual users' evolving understanding. Furthermore, errors in automatic extraction propagate through the structure without opportunities for user correction or refinement.

The progression from manual authoring to automatic construction to LLM-augmented exploration reflects a broader trajectory in knowledge representation systems [20, 40]. Manual visualization tools offer full user control but no intelligent assistance [9, 37]. Automatic knowledge graph systems leverage computational power but exclude human participation, producing structures that may not align with how users need to understand information [12, 34]. Recent LLM-augmented systems generate structured visualizations from LLM outputs but provide limited support for users to reshape the underlying knowledge structure.

Graphologue [21] converts LLM text responses into interactive node-link diagrams, enhancing comprehension of complex answers, but the generated structure remains a read-only output that users cannot refine to reflect their evolving understanding. Sensecape [51] supports multilevel exploration through hierarchical diagrams at different abstraction levels, yet the knowledge structure itself is system-determined. Selenite [29] addresses the cold-start problem in sensemaking by generating comprehensive overviews of options and criteria from LLMs, scaffolding exploration of unfamiliar domains. Luminate [50] takes a different approach by structuring the *design space* of LLM outputs, enabling users to explore diverse responses along meaningful dimensions rather than converging on a single answer. Dück et al. [13] support claim retrieval in large document corpora through multiple exploration pathways, combining keyword search with hypothesis-driven retrieval and consistency checking. Each of these systems advances a specific aspect of LLM-augmented knowledge exploration, yet the resulting representations are generated by the system and not persistently editable by users—users can navigate and query but cannot contribute their own knowledge to an evolving shared structure.

Commercial tools have also begun incorporating AI capabilities for knowledge work. NotebookLM[1] supports document-grounded question answering over user-uploaded sources and recently added AI-generated mind maps that visualize connections between source concepts. Notion AI[2] enables retrieval and content generation across a workspace of pages and databases. Miro AI[3] offers AI-assisted diagram generation on a collaborative visual canvas, while Excalidraw[4] and Lucidchart[5] support text-to-diagram conversion with AI features. Several of these tools already support forms of knowledge-level bidirectionality. NotebookLM and Notion AI, for example, let users both query and contribute to underlying knowledge stores. However, NotebookLM's core interaction is document-grounded question answering; mind maps are a secondary output of that process rather than the central artifact users build upon. Notion AI, similarly, operates entirely within a page-and-database structure with no visual knowledge graph at all. Contributions and retrievals thus remain distributed across separate views rather than integrated within a single visual structure. Research on knowledge visualization suggests this separation has cognitive consequences: Carneiro et al. [6] found that graph-based interfaces better support reasoning over non-linear structures than equivalent textual representations, Larkin and Simon [25] showed that spatial representations reduce cognitive effort by supporting perceptual inference, and Binks et al. [3] found that the cognitive value of knowledge representations lies in the transformations between them. Together, these findings suggest that integrating retrieval and contribution within a single visual knowledge structure may support reasoning that separate-pane designs do not afford. In such a structure, users can see how new content relates to existing knowledge and reorganize the whole.

On the technical side, Retrieval-Augmented Generation (RAG) [27] has become a foundational approach for grounding LLM responses in external knowledge bases, combining generative capabilities with document retrieval to improve factual accuracy [48]. Subsequent work has extended RAG through hierarchical retrieval strategies such as RAPTOR [45], which recursively clusters and summarizes text chunks to enable retrieval at varying levels of granularity. Multi-agent architectures [57] have further expanded the capabilities of LLM systems by coordinating specialized components for complex tasks. However, existing agent architectures are designed primarily for task completion rather than knowledge structure evolution [52, 53]—they treat the knowledge base as a static resource to be queried, not a shared artifact to be co-constructed. MindTrellis addresses the challenges with a multi-agent pipeline, enabling knowledge construction to be treated as collaborative rather than system-determined.

Across the research and commercial landscape surveyed above, research systems advance retrieval, visualization, or AI-assisted

[1] https://notebooklm.google.com
[2] https://www.notion.com/product/ai
[3] https://miro.com
[4] https://excalidraw.com
[5] https://www.lucidchart.com

exploration but produce structures users cannot reshape; commercial tools have begun supporting knowledge-level bidirectionality but distribute retrieval and contribution across separate views. The theoretical and empirical evidence on knowledge visualization suggests that spatially integrating both activities within a unified visual structure may yield cognitive benefits that separate-pane designs do not afford. MindTrellis pursues this direction, enabling users and AI to co-construct an evolving knowledge graph where both querying and contributing operate on the same structure grounded in source documents.

3 DESIGNING MINDTRELLIS

To understand user needs for knowledge building from multiple sources, we conducted a formative study that informed six key challenges; we then derived design goals that guided the development of MindTrellis.

3.1 Formative Study

We recruited six graduate students with diverse academic backgrounds (three female and three male) in computer science (two), neuroscience (two), and human-computer interaction (two). The study included a 20-minute knowledge exploration task where participants studied three psychology documents they had no prior familiarity with, followed by a 10-question open-book quiz testing their comprehension of the material. Participants first explored using only the three source documents without any AI assistance, then explored the same materials using a linear chatbot interface powered by a RAPTOR retriever (the same retriever used in our main user study). After completing both phases, we conducted semi-structured interviews covering their information-seeking strategies, challenges encountered, and preferences for tool support. The interviews were audio-recorded and transcribed for analysis. Two authors independently conducted initial open coding of the transcripts, then collaboratively aligned codes and constructed themes. We identified six key challenges that motivated our design.

C1: Navigating and synthesizing multiple documents is cognitively demanding. When exploring without AI assistance, participants found it tedious to locate and integrate information scattered across three separate documents. P6 described the experience as *"pretty tedious... I was slightly annoyed because I had to keep switching back and forth."* P2 noted the difficulty of tracking information across sources: *"I found it in one note but couldn't find the corresponding keyword"* in another. The cognitive burden of manually navigating, cross-referencing, and synthesizing content left participants fatigued and prone to missing relevant connections, motivating the need for AI-assisted tools.

C2: Connecting related concepts across sources remains difficult. Even with AI assistance, participants struggled to recognize when the same concept appeared across documents with different terminology. P3 observed that *"the wording is different, maybe they use different terminology to describe the same thing."* While the RAG-based chatbot could retrieve relevant passages, it did not explicitly surface connections across sources or help users reconcile different framings of the same concept. The burden of integration remained on users in both phases, who had to manually identify that two differently-worded passages referred to the same underlying idea.

C3: Retrieved content lacks structural organization for knowledge building. While the RAG-based chatbot reduced navigation burden by retrieving relevant information, participants found that linear text responses made it difficult to understand relationships between concepts. P1 requested *"a mind map"* because *"a pile of text is not as intuitive as a diagram."* P5 similarly suggested *"a mind map or matrix map to visualize the information,"* and P3 wanted *"an outline with a list of keywords that lets me navigate back to the original content."* These comments reveal a mismatch between how conversational LLM interfaces present information and how users naturally organize knowledge: users need to see not only individual facts but also how concepts relate hierarchically and semantically.

C4: LLM-generated content poses trust and verification concerns. While participants appreciated RAG-assisted retrieval efficiency, many hesitated to trust responses without verification. P4 articulated this tension: *"the problem with AI tools is for academia. I cannot see from where that content is coming."* P3 noted that for important tasks, they *"might even want to use backtracking for every question to go back to the source and make sure it didn't hallucinate."* P6 felt *"less confident compared to the notes. I'm worried if there will be hallucination."* Without clear provenance linking LLM-generated content to source materials, users face an uncomfortable choice between efficiency (accepting AI output) and accuracy (manually verifying everything).

C5: Exploration does not culminate in an evolving knowledge structure. When using the chatbot, participants found that interactions were transient; each query produced an independent response with no persistent structure building over time. P6 noted that the chatbot *"doesn't remember the conversation history"* and wished follow-up questions could build upon previous context. P3 described a similar issue with retention: after receiving responses from the chatbot, P3 reflected *"I don't remember anything,"* suggesting isolated responses fail to support lasting knowledge construction. Effective knowledge building is cumulative: users progressively expand understanding by building on prior knowledge and constructing comprehensive mental models. When each interaction exists in isolation, users cannot see their exploration history or develop a coherent overall structure.

C6: AI-generated structures are static and non-modifiable. Even when LLM provides organized output, participants wanted to reshape it according to their own understanding, but found existing systems did not support such customization. P5 emphasized that *"this kind of exploration is very personal"* and *"the logic follows how you understand the thing,"* expressing desire to *"customize"* how information is structured. P1 similarly wished to *"export a document... then iterate and update"* the content. Knowledge building is inherently personal and iterative: users develop understanding by actively reorganizing and refining information structures. Systems producing static, non-editable output prevent users from engaging in this essential process.

3.2 Design Goals

Drawing from the challenges identified in our formative study (C1–C6), we established the following design goals to guide the development of MindTrellis.

D1: Represent multi-sourced knowledge through a co-created, visually-structured graph (C1, C2, C3). A co-created knowledge graph combines the hierarchical organization of mind maps with the semantic relationships of concept maps [44], enabling users to see how concepts connect in an intuitive, layered format. This representation makes structural relationships explicit and visible, reducing the cognitive effort of navigating multiple sources (C1) and understanding conceptual connections (C3). Integrating information from multiple sources into a unified visual structure also helps users identify cross-document connections that might otherwise remain hidden (C2). The knowledge graph is co-created in the sense that AI provides intelligent assistance in organizing and connecting information, while users retain the ability to shape and refine the structure according to their understanding.

D2: Enable bidirectional interaction for cumulative knowledge building (C5, C6). Bidirectional interaction allows users to both query from and contribute to the knowledge structure [1, 2]. In the query direction, users retrieve information grounded in their documents, with responses organized within the evolving knowledge graph. In the contribution direction, users actively shape the structure by adding concepts, modifying relationships, and reorganizing hierarchies to reflect their developing understanding. This transforms users from passive consumers into active co-creators. Contributions persist and accumulate [35], allowing the knowledge graph to evolve continuously as users explore and refine their mental models (C5). Supporting modification of AI-generated structures enables the personal, iterative refinement essential to deep knowledge building (C6).

D3: Support flexible interaction with transparent provenance (C4). Two complementary interaction modes accommodate diverse preferences and task demands [8]. Natural language interaction through a chat interface enables conversational queries and structural modification commands. Direct manipulation on a visual canvas [19, 33] provides precise control through clicking, dragging, and toolbar operations. Natural language is efficient for complex queries and bulk operations; direct manipulation offers spatial understanding and fine-grained control. Regardless of interaction mode, transparent provenance allows users to trace any information back to its source document, directly addressing trust concerns by supporting verification when needed (C4).

4 MINDTRELLIS

Guided by the design goals established in Sec. 3.2, we developed MindTrellis to enable human-AI collaborative knowledge construction. Achieving this goal requires addressing three technical challenges: first, user input is inherently ambiguous, as the same natural language statement could be a query seeking information or a contribution intended to modify the knowledge structure; second, users navigating unfamiliar topics need information at varying levels of granularity; and third, when users contribute new knowledge, the system must determine appropriate placement within the existing structure while maintaining coherence.

To address these challenges, we developed a multi-agent architecture that coordinates specialized components: an Oracle for intent classification, an Adaptive Retriever that supports variable-granularity retrieval, and a Map Manager for coherent knowledge placement. These components operate on a shared knowledge graph that evolves through user interaction.

4.1 System Overview

MindTrellis enables users to both retrieve information from and contribute insights to an evolving knowledge graph through two complementary pathways (Fig. 1, Fig. 3). In the **query pathway**, users pose questions to explore the knowledge base. The system searches for relevant information, synthesizes a response grounded in the uploaded documents, and optionally expands the knowledge graph with newly retrieved content. This pathway supports exploratory information seeking, allowing users to progressively deepen their understanding of unfamiliar topics (**D1**, **D2**). In the **contribution pathway**, users add new concepts, modify existing relationships, or reorganize the knowledge structure to reflect their developing understanding. The system interprets these contributions, determines appropriate placement within the existing hierarchy, and integrates the changes while maintaining structural coherence. This pathway enables users to externalize their insights and shape the knowledge representation according to their own mental models (**D2**, **D3**). These two pathways interact with a shared **knowledge graph** that combines the hierarchical structure of mind maps with the semantic relationships of concept maps. The graph evolves continuously as users query and contribute, capturing both the information retrieved from documents and the organizational decisions made by users. This co-created representation accumulates the user's explorations and serves as a lasting artifact that reflects their learning journey (**D1**).

To handle the complexity of supporting both pathways, MindTrellis employs a **multi-agent pipeline** consisting of three specialized components: the Oracle coordinates user interactions and classifies intent; the Adaptive Retriever searches the knowledge base at varying levels of granularity; and the Map Manager interprets contribution commands and executes modifications to the graph. These components collaborate to provide seamless bidirectional interaction, detailed in Sec. 4.4.

Users engage with MindTrellis through two complementary interaction techniques: **direct manipulation** on the canvas (Fig. 2f), where users can create nodes, drag connections, and reorganize the layout; and **natural language** in the chat panel (Fig. 2g), where users can issue queries, request expansions, or specify modifications through conversational commands. This combination accommodates diverse interaction preferences while supporting both exploration and contribution (**D3**), detailed in Sec. 4.5.

4.2 User Scenario

Before detailing the technical components, we present a user scenario to demonstrate how MindTrellis supports knowledge exploration and construction. Suppose Robert is an undergraduate student preparing for a course on climate change. With no prior knowledge of the subject, he uploads documents and readings shared by his professor and uses MindTrellis to explore the key concepts.

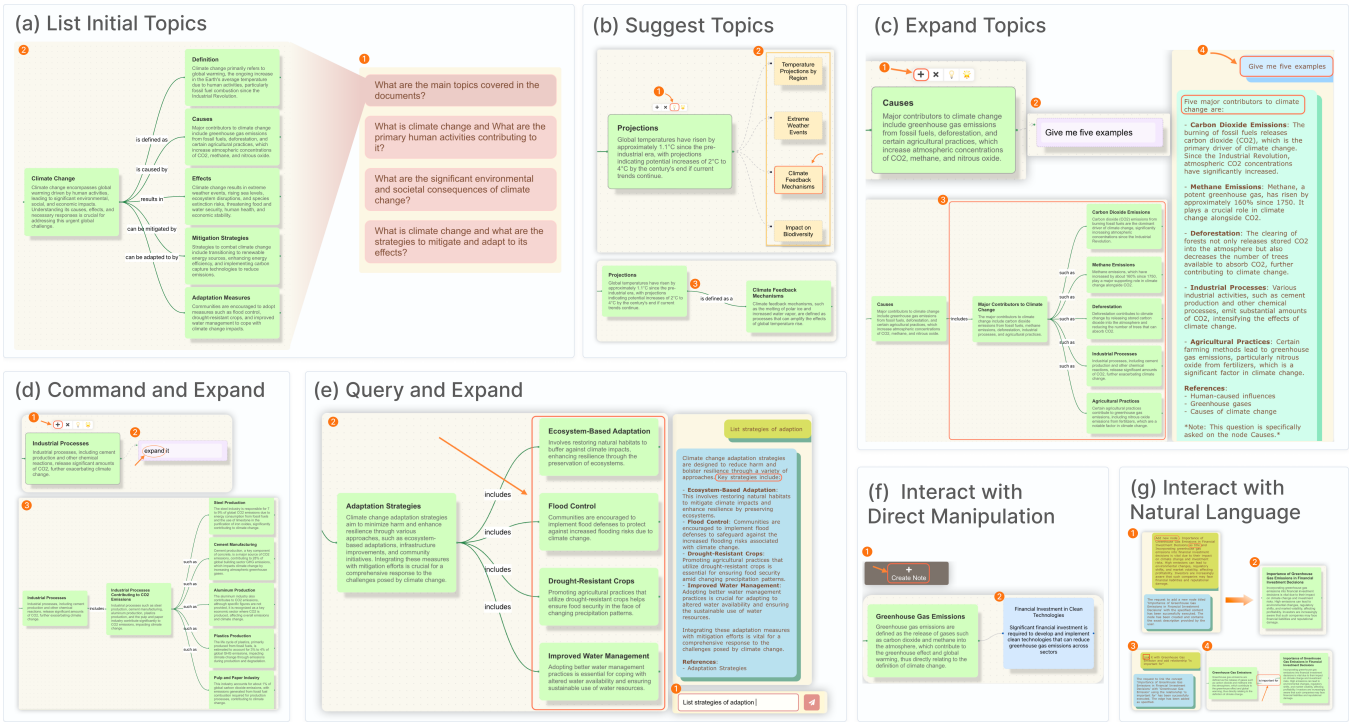


Figure 2: Key functionalities of MindTrellis demonstrated through the user scenario. (a) Initial exploration generating a hierarchical overview of climate change concepts. (b) System-generated suggestions for expanding the "Projections" node. (c) Query-driven expansion requesting examples of climate change causes. (d) Command-based expansion of industrial processes. (e) Chat-based query adding adaptation strategies. (f) Direct manipulation: creating a custom node and linking it to existing concepts. (g) Natural language contribution: adding a new node and defining its relationship through text commands.

Uncertainty About Initial Questions. At the outset, Robert feels overwhelmed by the unfamiliar content and is unsure where to begin (**C1**). He selects one of the suggested questions in the chat panel: "What are the main topics covered in the documents?" The system generates an initial node diagram that visually represents key topics in a hierarchical structure, with labeled edges indicating relationships between concepts. Each node includes a title and additional details for further exploration (Fig. 2a).

As Robert reviews the graph, he becomes curious about "Projections" but is uncertain about its subtopics. He clicks the suggestion button 💡 on the node toolbar, which provides four options for related topics (Fig. 2-b2). Intrigued by "climate feedback mechanisms," he selects it, and the system adds a new node linked to "Projections" with an edge indicating that climate feedback mechanisms are a component of climate projections (Fig. 2-b3). This feature allows Robert to expand the graph via system-generated suggestions, even when he lacks familiarity with the content (**C1**, **D2**).

Knowing What to Ask. Robert decides to learn about the causes of climate change. He clicks the plus icon + on the "Causes" node and types "Give me five examples." The system adds a new node titled "Major Contributions to Climate Change" with the relationship "includes," and five examples such as "Deforestation," appear as child nodes (Fig. 2c). The question and response are also recorded in the chat panel for reference (Fig. 2-c4).

One example, "Industrial Processes," catches Robert's interest. He types "expand this," and the system adds detailed nodes about specific industrial processes contributing to CO2 emissions (Fig. 2d). Next, Robert types "List strategies of adaptation" into the chat, and the system responds with relevant strategies, adding them as child nodes under "Adaptation Strategies" (Fig. 2e). By progressively exploring topics, Robert continues to expand his understanding. The hierarchical structure allows Robert to navigate complex concepts with ease, maintaining a clear overview of related information. Unlike traditional workflows that require switching between a chat interface and a separate mapping tool (**C5**), MindTrellis integrates exploration and visualization in a unified environment (**D1**, **D2**).

Contributing External Knowledge. Robert finds video content on climate change and its business implications that he wants to integrate into his exploration. In conventional systems, users cannot modify the underlying knowledge base (**C3**); they can only retrieve information but not contribute their own insights. With MindTrellis, Robert creates a custom node using the toolbar button + for "financial investment in clean technologies" and links it to "Greenhouse Gas Emissions" (Fig. 2f), demonstrating direct manipulation to add new content (**D3**).

He then wants to incorporate another insight from the video: that emissions data increasingly influences investment decisions. Rather than manually locating where this concept belongs in the growing graph (**C6**), Robert simply types into the chat: "Greenhouse

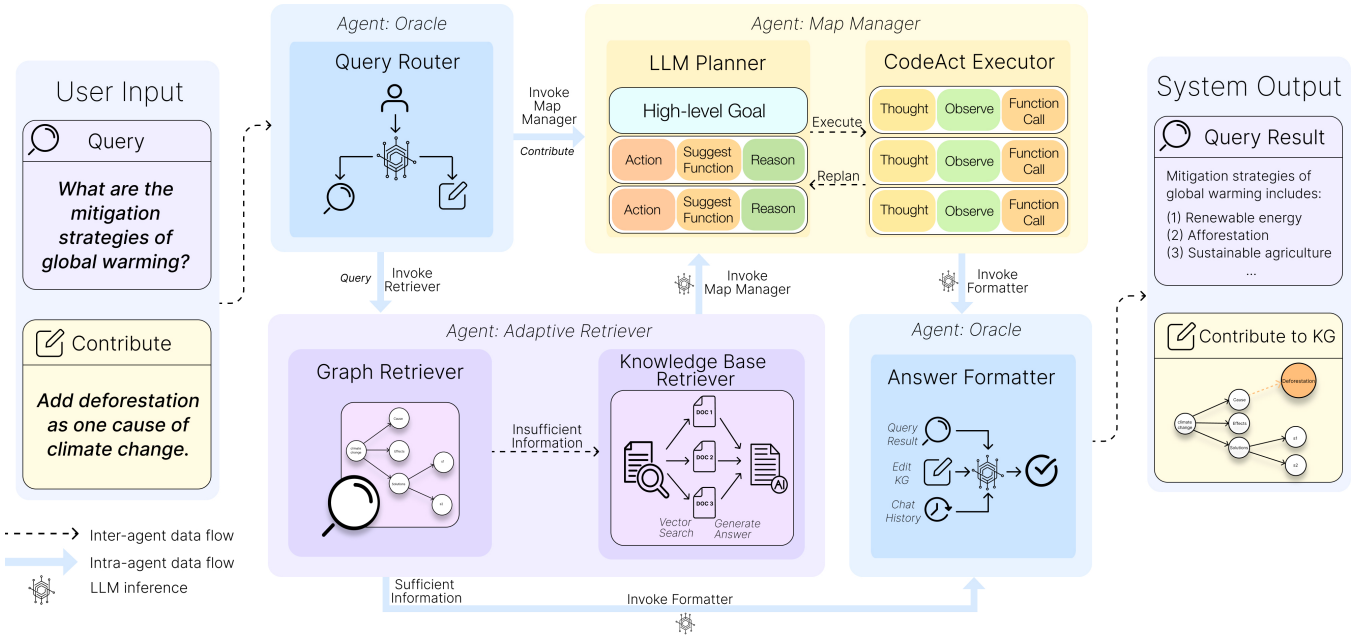


Figure 3: Overview of the MindTrellis multi-agent pipeline supporting bidirectional interaction. User input is routed by the Oracle to either the Query Pathway (handled by the Adaptive Retriever) or the Contribution Pathway (handled by the Map Manager). Both pathways interact with the co-created knowledge graph, which evolves through iterative user engagement. The Oracle coordinates responses and maintains conversational context across interactions.

gas emissions are becoming important for financial investment decisions," followed by a detailed explanation. The system recognizes this as a contribution rather than a query, analyzes the existing graph structure to identify "Greenhouse Gas Emissions" as the relevant anchor node, creates a new node capturing the concept, and establishes a relationship labeled "is important for"—all without requiring Robert to specify commands or node names (Fig. 2-g).

When Robert reviews the result, he notices the system placed the new node as a child of "Greenhouse Gas Emissions." He prefers it as a sibling node instead, so he types "move this node to be at the same level as Greenhouse Gas Emissions." The Map Manager re-plans the graph structure and executes the modification, demonstrating how users can refine the system's intelligent placement decisions through natural conversation.

This interaction illustrates the agentic nature of the contribution pathway: users express their knowledge naturally, and the system handles intent classification, anchor identification, relationship inference, and structural placement, transforming conversational input into coherent graph modifications (**C3**, **C6**, **D2**, **D3**).

Revisiting Previous Explorations. Later, Robert wants to revisit information about "carbon sinks" but cannot locate it in the expanded graph (**C2**, **C4**). He types "What are carbon sinks?" into the chat. Since this information already exists in the graph, the system responds without creating a duplicate node, helping Robert access the information without redundancy.

By the end of his session, Robert had built a well-structured representation of climate change concepts using MindTrellis. The system allowed him to explore relationships between topics while organizing information into a coherent structure that serves as a valuable reference for his course.

4.3 Knowledge Representation for Human-AI Co-Creation

We design the knowledge graph in MindTrellis to serve dual purposes: it should be intuitive for users to navigate and understand, while also being structured enough for the system to manipulate coherently when integrating user contributions. To achieve this, we integrate features from both mind maps and concept maps. This combined representation supports human comprehension through familiar hierarchical organization, while enabling AI-assisted construction through explicit semantic structure (**D1**).

From **mind maps**, the knowledge graph inherits a hierarchical, top-down approach that allows users to visualize relationships between topics clearly. Nodes represent concepts, and parent-child links depict hierarchical relationships, creating a natural exploration path from general topics to specific details.

From **concept maps**, the knowledge graph incorporates semantic edge labels that provide context on how concepts relate. For example, when exploring climate change, a user might encounter the node "Greenhouse Gas Emissions" connected to "Fossil Fuel Combustion" with an edge labeled "is caused by," and to "Carbon Pricing Policies" with an edge labeled "can be mitigated through." These labels allow users to grasp the nature of relationships at a glance without reading detailed content.

The graph also supports *common child nodes* that connect to multiple parents, representing cross-cutting concepts relevant to several categories. For instance, "Deforestation" relates to both

"Causes of Climate Change" (with relationship "contributes to") and "Biodiversity Loss" (with relationship "leads to"), capturing how a single phenomenon connects to multiple broader themes without duplicating the node.

This combination enables effective co-creation: the hierarchical structure helps both users and the system navigate the graph and identify appropriate locations for new content, while semantic labels help both parties understand relationships: users can quickly comprehend the knowledge structure, and the system can leverage semantics to suggest meaningful placements for contributions.

Through iterative user interaction, the knowledge graph evolves progressively and captures the user's developing understanding (**D2**). Rather than presenting all information at once, the system expands the graph incrementally in response to user queries and contributions. Such incremental expansion aligns with exploratory information seeking, where users prefer to encounter information gradually as their familiarity grows, preventing cognitive overload while supporting continuous discovery.

4.4 Multi-Agent Pipeline for Bidirectional Interaction

To address the challenges of intent ambiguity, granularity matching, and coherent placement, we developed a multi-agent pipeline comprising three specialized components: the Oracle for intent classification and routing, the Adaptive Retriever for variable-granularity knowledge retrieval, and the Map Manager for coherent knowledge placement (Fig. 3).

Oracle: Intent Classification and Routing. The Oracle serves as the central coordinator that classifies user intent and routes requests to the appropriate pathway. It analyzes each input with a large language model to determine whether the input represents a *query* (seeking information) or a *contribution* (modifying the knowledge graph). The classification considers linguistic cues, including imperative structures and explicit commands like "add" or "link" suggest contributions, while interrogative forms suggest queries, as well as conversational context from previous interactions. For queries, the Oracle invokes the Adaptive Retriever and formats responses for the user. For contributions, it directs the request to the Map Manager and reports the outcome.

Adaptive Retriever: Variable-Granularity Knowledge Retrieval. Users exploring unfamiliar topics need information at different levels of detail—broad overviews when orienting themselves, specific facts when investigating particular concepts. Standard retrieval-augmented generation (RAG) approaches retrieve a fixed number of text chunks, which proves insufficient for summative questions requiring broader context and provides no mechanism for adapting to user familiarity.

To address this, we employ a Raptor retriever [45] that enables retrieval at varying granularities. The retriever recursively embeds, clusters, and summarizes text chunks into a hierarchical tree structure. When users are new to a topic, the retriever draws from higher levels of the tree to provide broader overviews. As users become more familiar, it retrieves from lower levels to supply detailed, grounded facts.

The retrieval pipeline (Fig. 4) includes four stages: a *Query Rewriter* optimizes the user's input for vector database retrieval;

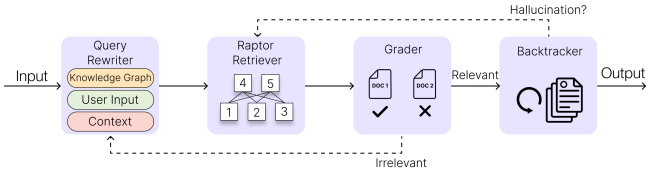


Figure 4: The Adaptive Retriever pipeline consists of four distinct stages: (i) Query Rewriter: refines the user's input to optimize it for retrieval, (ii) Raptor Retriever: retrieves relevant text chunks at various levels of granularity by recursively embedding and clustering information, (iii) Text Chunks Grader: evaluates the retrieved chunks for relevance to the user's query, filtering out less pertinent content, and (iv) Backtracker: links the generated response back to the original source, providing metadata to ensure transparency and grounding of information.

the *Raptor Retriever* searches for relevant chunks at appropriate granularity; a *Grader* evaluates retrieved chunks for relevance and filters out tangential content; and a *Backtracker* links responses to source documents with metadata (title, page number) to ensure transparency and reduce hallucination risk.

Map Manager: Coherent Knowledge Placement. When users contribute new knowledge, the system needs to interpret their intent and integrate the contribution coherently. A node about "deforestation" should connect to "causes of climate change" with an appropriate relationship label, not appear as an orphan or attach to an unrelated concept.

The Map Manager handles contribution processing through a plan-and-execute scheme with iterative error correction (Algorithm 1). In the *planning* phase, an LLM interprets the user's objective, analyzes the current graph state, and generates a sequence of function calls to achieve the goal. The knowledge graph is represented internally as an adjacency matrix, and the planner produces operations such as `AddNode()`, `AddEdge()`, `DeleteNode()`, or `UpdateNode()` with appropriate arguments.

In the *execution* phase, a CodeAct agent [55] executes each planned operation sequentially. If an operation fails (e.g., target node not found, invalid argument format), the executor attempts self-correction by reasoning from the error message and conversation context. If self-correction fails after N attempts (we use $N = 5$), the executor reports the error back to the planner, which replans based on the current graph state and error information. This plan-execute-replan cycle continues until all operations complete successfully or the maximum recursion depth is reached.

Consider a user who inputs "Greenhouse gas emissions are becoming important for financial investment decisions." The Oracle classifies this as a contribution based on its declarative structure expressing a relationship. The Map Manager's planner analyzes the graph, identifies "Greenhouse Gas Emissions" as the anchor node, and generates two operations: creating a new node titled "Financial Investment Relevance" and establishing an edge labeled "is important for." The executor runs each operation; if, for instance, the edge creation fails because the relationship type is not recognized, the self-correction mechanism reasons from the error and retries

Algorithm 1: Map Manager: Plan-and-Execute with Self-Correction

Input : u: user input; G: current graph state; H: conversation history
Output : G': updated graph; r: response to user

```
depth ← 0
while depth < MAX_DEPTH do
    ops ← Plan(u, G, H)        // e.g., [AddNode(...), AddEdge(...)]
    foreach op ∈ ops do
        for attempts ← 0 to N do
            result ← Execute(op, G)
            if result.success then G ← result.newState; break
            else op ← SelfCorrect(op, result.error, H)
        if attempts = N then H ← H ∪ {error}; depth++; goto line 3
    return G, FormatResponse(ops)
return G, "Max attempts reached."
```

with a corrected argument (e.g., changing to "influences"). Upon successful completion, the Oracle confirms the update.

This architecture provides several advantages over single-prompt approaches: specialized components can be refined independently, error handling is more robust through the plan-execute-replan cycle, and the separation between intent classification, retrieval, and graph modification enables cleaner reasoning at each stage. Detailed prompts for each component are provided in Appendix A.

4.5 User Interface

The MindTrellis interface comprises three main components (Fig. 1): a file panel for document management, a canvas for knowledge graph visualization and direct manipulation, and a chat panel for natural language interaction.

File Section. The file panel (Fig. 1a) allows users to upload and organize documents for their knowledge base. Users can create folders by clicking the folder icon and dragging files into the desired location. The system automatically processes uploaded documents and stores them in the vector database for retrieval.

Chat Section. The chat panel (Fig. 1c) supports natural language queries and contributions. At the start of a session, the system provides suggested questions to help users begin exploration. Users can type queries to retrieve information (e.g., "What are the causes of climate change?") or issue commands to modify the knowledge graph (e.g., "expand this," "delete," "link X to Y"). Each response appears in the chat along with source references, and corresponding changes are reflected on the canvas immediately.

Widget Bar. The widget bar at the top of the interface provides quick access to key functions. The "File Manager" and "Chat with AI" buttons toggle the side panels. The "Create Node" button allows users to add custom nodes for notes or external knowledge. The "Group Nodes" button enables users to cluster related nodes; once grouped, users can ungroup them or remove individual nodes with the scissor icon . The "Rearrange Canvas" button automatically reorganizes the layout when the graph becomes cluttered.

Canvas Section. The canvas (Fig. 1b) displays the knowledge graph and supports direct manipulation for both exploration and contribution. When a user selects a node, a toolbar appears with four functions: the plus icon opens an input box for queries focused on that node; the delete icon removes the node; the suggestion icon provides expansion recommendations; and the star icon highlights important nodes in yellow.

The canvas also features *semantic zoom* to help users navigate large graphs (**D1**). At a zoomed-out level, only node titles are visible, which provides a high-level structural overview. As users zoom in, detailed content within each node becomes readable, allowing fluid transitions between broad orientation and focused exploration. The graph uses a horizontal hierarchical layout based on breadth-first traversal, with root nodes on the left and child nodes extending rightward. Users can drag nodes to override computed positions; the "Rearrange Canvas" button recomputes the full layout from the current graph topology when the arrangement becomes cluttered.

4.6 Implementation Details

MindTrellis consists of a React.js[6] frontend and a Flask backend. For language model inference, we use GPT-4o[7] for planning tasks in the Oracle and Map Manager, and GPT-4o-mini for simpler tasks to reduce latency. The backend workflow is implemented using LangGraph [24], which abstracts the multi-agent coordination as a state machine. For the Adaptive Retriever, we use LangChain's ChromaDB wrapper[8] to construct the vector store and OpenAI's text-embedding-3-small[9] for computing embeddings. The knowledge graph data structure was implemented from scratch to support the hierarchical and semantic features described in Section 4.3. For the user study, we deployed MindTrellis on Microsoft Azure.

5 USER STUDY

We conducted a controlled user study to evaluate MindTrellis's effectiveness in supporting users' knowledge management and information seeking by comparing against a baseline.

5.1 Participants

We recruited 12 participants (6 female, 6 male, aged 18-35), including 5 Master's students, 4 PhD students, and 3 undergraduates. They had diverse backgrounds spanning computer science, neuroscience, data science, and human-computer interaction. All participants had prior experience with generative AI tools such as ChatGPT and Gemini, and most reported using these tools frequently. Additionally, the majority had experience using mind maps for organizing or learning information. The experiments were performed on a laptop computer equipped with a mouse. Each participant received $30 for their participation.

[6] https://react.dev
[7] https://platform.openai.com/docs/models/gpt-4o
[8] https://python.langchain.com/docs/integrations/vectorstores/chroma/
[9] https://platform.openai.com/docs/models/text-embedding-3-small

5.2 Design

We employed a within-subject approach to compare two systems, MindTrellis and a baseline, by curating two study tasks over two different datasets, using a counterbalanced design.

Datasets. The tasks required participants to create a slide deck by using the systems to explore and understand the concepts from the datasets. We selected datasets covering two distinct domains: climate change and AI ethics. For each topic, we compiled a collection of related documents sourced from Wikipedia. The climate change dataset consisted of six documents, while the AI ethics dataset included five documents. Each document contained between 2,000 and 4,000 words. To ensure compatibility with our text-only retrieval system, we removed all figures from the articles, as our current RAG implementation only supports textual content.

Baseline. Our baseline was a retrieval-only system with graph visualization capabilities (see Appendix B). The baseline consisted of two components: the same Adaptive Raptor Retriever used in MindTrellis for querying the document corpus, and a visualization interface that automatically converts LLM responses into static node-link diagrams. Participants first selected a dataset, then queried it through the retriever interface. After receiving answers, they could transfer the results to the visualization interface, which generated corresponding concept maps. Participants could then ask additional questions and examine the visualization.

We chose this baseline to isolate the effect of the contribution pathway, as both systems support retrieval and visualization, but only MindTrellis allows users to contribute to and modify the knowledge structure. This design choice enables us to evaluate whether bidirectional interaction (querying *and* contributing) improves knowledge organization compared to retrieval-only interaction. We acknowledge that comparing systems with different capabilities introduces potential confounds; however, our goal was to evaluate whether the contribution pathway improves knowledge organization, which required a baseline without this capability.

5.3 Procedure

The study lasted approximately two hours for each participant, consisting of the following steps.

Introduction and Consent (15 minutes). Initially, we provided a brief introduction to the study background, goals, and the two systems, and administered a demographics pre-questionnaire.

System Tutorials (10 minutes each). Before using each system, we provided an instructional video that introduced its key features, giving participants a basic understanding of the interface and functionality. Participants were then given time to interact with the system they would use for the upcoming task. During this time, we guided them through the features, showing them how to explore and build knowledge graphs. A test dataset on the topic of operating systems was used during this familiarization phase.

Study Tasks (25 minutes each). A Latin square counterbalancing method was used to alternate between systems and datasets across participants. Participants were told: "Imagine you have been invited to give a presentation on the selected topics. You will use Google Slides to create slide decks on selected topics. Each slide deck should be created by exploring and understanding the content provided in the documents using the given system. You do not need to add images or any other visual decorations. You will have 20 minutes to work with each system." The participants' interactions with the systems, as well as their process of building the slide decks, were recorded.

Post-questionnaire and Interview (20 minutes). After each task, participants completed a post-questionnaire to evaluate the usefulness and effectiveness of the systems, along with specific feature ratings for MindTrellis. At the end of the study, we conducted a 10-15 minute semi-structured interview where participants compared the two systems based on their experiences. They also provided insights on specific features of MindTrellis.

6 USER STUDY RESULTS

We analyzed the results using a mixed-method approach that included system usage logs, post-questionnaires, and semi-structured interviews. We compared MindTrellis with the baseline across usability, effectiveness of the generated knowledge graphs, and support for knowledge exploration using the Wilcoxon signed-rank test (Fig. 5: Q1–11 and Fig. 6: Q12–16). Below, we report findings organized by our three design goals (D1–D3).

6.1 Co-Created Knowledge Graph Supports Exploration (D1)

Participants reported that the co-created knowledge graph enhanced their exploration by providing a structure they could shape together with AI assistance. The co-created graph received higher ratings for organizing information (Q3: $Mdn_M = 7.0 > Mdn_B = 4.5$, $p < 0.01$, $r = 0.883$), ease of use (Q4: $Mdn_M = 7.0 > Mdn_B = 4.5$, $p < 0.01$, $r = 0.883$), and supporting topic understanding (Q5: $Mdn_M = 7.0 > Mdn_B = 5.0$, $p < 0.01$, $r = 0.883$).

The co-created graph supported exploration in three main ways. First, participants found that being able to shape the visual organization made material "easier to comprehend" (P4) and allowed them to "organize [the] knowledge graph fully before starting on slides" (P1). P12 appreciated how the system "allows me to control the flow of the graph that follows my own mindset," emphasizing the collaborative nature of the representation. Second, the structure enabled in-depth exploration while maintaining coherence. P8 could focus on "higher-level ideas" and P9 could "expand on key aspects of a topic in a consistent manner" (Q9: $Mdn_M = 4.0 > Mdn_B = 3.0$, $p < 0.05$, $r = 0.883$). Third, detailed content with titles, explanations, and semantic edge labels simplified comprehension. P2 noted that seeing "detailed relationships between nodes [on edges]" enabled deeper understanding, and P5 found that "relevant examples" made complex topics "simpler and more straightforward."

In contrast, the baseline generated graphs automatically without user input, producing structures that participants found difficult to work with. P2 described the baseline as offering only "vague relationships," and P9 found the information "scattered," making it "challenging to maintain a structured overview." P1 wanted "the ability to organize" when using the baseline, and P5 noted that "the space is really limited, and I couldn't freely zoom in or zoom out to see the overview." Unlike the baseline's fixed, auto-generated structure, the co-created graph introduces information progressively as users explore, reducing initial overwhelm (Q1: $Mdn_M = 1.0 < Mdn_B = 3.5$, $p < 0.01$, $r = 0.815$). Participants also reported that MindTrellis

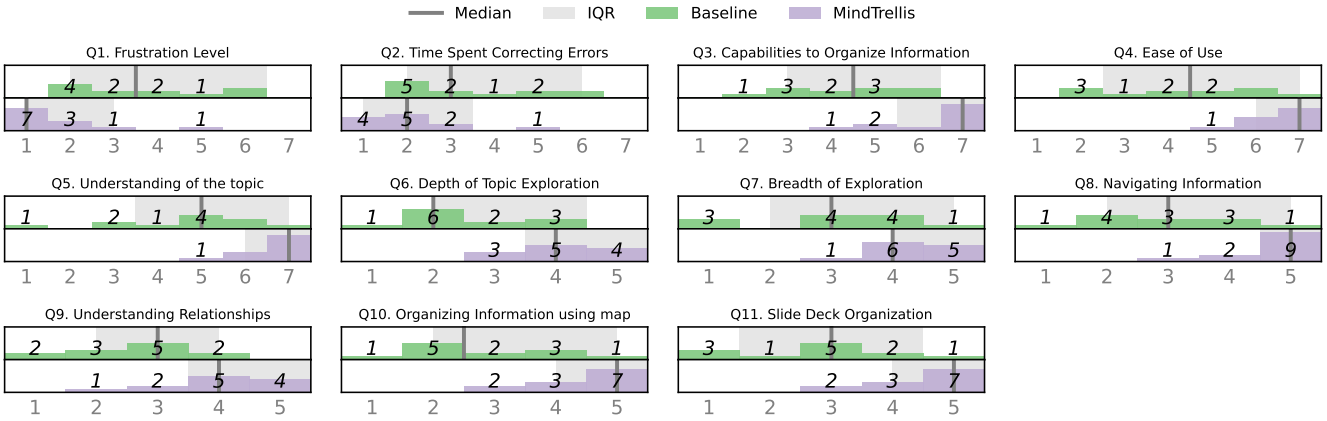


Figure 5: Participants' ratings on usability (UMUX), task support, and depth and breadth of information exploration. Usability measures (Q1–5) were rated on a 7-point Likert scale, assessing whether MindTrellis met their requirements, ease of use, and frustration experienced. Task-related questions (Q6–11) were rated on a 5-point Likert scale, assessing how well the system supported understanding of the topic, the detail and breadth of information provided, and the system's capability in organizing slide deck content. For Q1 and Q2, lower scores are better; for the remaining questions, higher scores are better.

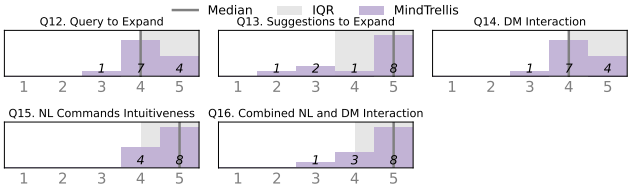


Figure 6: Participants' ratings on MindTrellis's effectiveness of knowledge expansion, system suggestions, and user interaction methods (direct manipulation and natural language commands). Ratings were collected on a 5-point Likert scale.

better supported creating "in-depth slides and connections across topics" (P3) (Q11: $Mdn_M = 5.0 > Mdn_B = 3.0$, $p < 0.01$, $r = 0.883$). These observations confirm the success of D1 in supporting exploration through a co-created, visually structured knowledge graph.

6.2 Bidirectional Interaction Enables Cumulative Knowledge Building (D2)

A key capability of MindTrellis is bidirectional interaction: users can both query from and contribute to the evolving knowledge structure. This section reports how participants engaged with both pathways and how the iterative cycle between them supported cumulative knowledge building.

6.2.1 *Query Pathway: Retrieving and Exploring Information.* Participants used the query pathway to retrieve information from their uploaded documents in four ways. First, they asked questions to retrieve information directly. P7 noted that "I could just type my question and get relevant information without searching through all the documents myself." Second, they issued commands to expand specific nodes. P2 "liked the option to extend nodes using natural language" because it allowed them to "expand on ideas directly and get more specific examples" (Q12: $Mdn = 4.0$, $IQR = 1.0$). Third, they requested context about specific nodes without losing the overall structure. P8 shared that "it allowed me to engage with specific nodes, which made learning about each topic more in-depth as I could explore related concepts seamlessly" (Q15: $Mdn = 5.0$, $IQR = 1.0$). Unlike the baseline, which changed the graph structure upon additional operations, MindTrellis preserved context during querying (Q2: $Mdn_M = 2.0 < Mdn_B = 3.0$, $p < 0.05$, $r = 0.758$). Fourth, they used the suggestion feature for targeted inspiration. P11 appreciated that "it is helpful when I'm new to a topic and don't know what should be included next. It gives me a direction to expand" (Q13: $Mdn = 5.0$, $IQR = 1.25$). P10 could "quickly explore and expand the knowledge graph without having to manually add nodes," and P6 noted efficiency gains: "it allowed me to see suggested connections and explore those connections quickly."

6.2.2 *Contribute Pathway: Adding and Refining Knowledge.* Beyond retrieval, participants actively contributed their own knowledge in two main ways.

First, when they had relevant domain expertise or prior knowledge, participants added new concepts that extended beyond the source documents. P12, a participant with a finance background, discovered connections between climate change and financial investment. They supplemented the knowledge base with their domain expertise: "I'm happy it made my slide more concrete." Similarly, P3 added concepts from prior coursework: "I added some nodes based on what I already knew, which helped me connect the new material to my existing understanding."

Second, when the system's organization did not match their mental models, participants reshaped the structure. For fine-grained adjustments, participants used direct manipulation to drag nodes and visually group concepts. P11 emphasized: "It gives me control over how I explore and arrange the information, which is essential for structuring my thoughts" (Q14). However, P9 noted that "expanding groups together would be a great addition," suggesting that batch operations on grouped nodes could further enhance direct manipulation. For bulk modifications—such as reorganizing entire sections or modifying multiple edge labels—natural language

proved more efficient. P10 noted that "typing a command was faster than dragging things around."

6.2.3 *Cumulative Building Through Iterative Interaction.* Participants fluidly moved between querying and contributing, and this iterative cycle enabled cumulative knowledge building. Rather than completing all queries before contributing or vice versa, participants alternated both pathways throughout their sessions.

Two dominant patterns emerged. In the first pattern, participants started with queries and transitioned to contribution when they identified gaps. P4 described this cycle: "I can keep asking questions, and they can keep inspiring me with new topics. It is easier to explore and expand on topics cohesively." For example, P12, a participant with a finance background, was exploring climate change documents to prepare a presentation. While reviewing content on greenhouse gas emissions, they recognized connections to their prior expertise—specifically, how emissions data relates to financial investment decisions such as ESG (environmental, social, governance) investing. Although the source documents did not cover this angle, they supplemented the knowledge base with their domain knowledge: "I'm happy it made my slide more concrete." They then queried further to expand those newly added concepts, building out the financial investment branch of the graph.

In the second pattern, participants contributed first and used queries to expand from their additions. P3 noted: "I added some nodes based on what I already knew, then asked the system to give me more details about those areas." This contribute-then-query approach allowed participants to anchor exploration in their existing understanding, using the system to fill in details around concepts they introduced.

The suggestion feature often triggered transitions between pathways. P5 observed: "Combining features like the suggestion tool and chat allowed me to explore the material in more depth." When suggestions surfaced unexpected connections, participants would contribute additional context from their knowledge, then query to explore those connections further.

Individual differences emerged in how participants balanced the two pathways. Some primarily used queries, relying on chat and suggestions to drive exploration. Others were heavily engaged with contribution, creating custom nodes and reorganizing extensively. P3 noted: "I spent a lot of time rearranging nodes and adding my own ideas because I wanted the graph to reflect how I think about the topic." This variance suggests that bidirectional interaction accommodates diverse knowledge-building styles. These findings confirm D2's goal of enabling cumulative knowledge building through bidirectional interaction.

6.3 Flexible Interaction with Transparent Provenance (D3)

Participants alternated between natural language and direct manipulation, leveraging each mode's strengths for different task demands. Transparent provenance enabled verification of retrieved information when needed.

A common pattern involved using natural language to add content, then switching to direct manipulation to adjust positioning. P9 described: "I would ask the system to add information, then drag the nodes around to organize them the way I wanted." Participants appreciated having both options available, choosing natural language for efficiency when precise placement was not critical, and direct manipulation when layout mattered (Q14: $Mdn = 4.0$, $IQR = 1.0$; Q15: $Mdn = 5.0$, $IQR = 1.0$; Q16: $Mdn = 5.0$, $IQR = 1.0$).

Participants also valued being able to trace retrieved information back to source documents. P8 mentioned that "knowing the answers came from my own documents made me more confident in the information," and P11 appreciated being able to "click on the citation and see exactly where the information came from." P3 added that "when I wasn't sure about something, I could always go back to the original document to check." However, P4 noted that "sometimes the system didn't quite understand what I was asking for, and I had to rephrase my question," suggesting opportunities for improving query interpretation.

These findings confirm D3's support for flexible interaction through complementary modes, with transparent provenance enabling verification and trust in the co-created knowledge graph.

7 PIPELINE EVALUATION

In addition to evaluating the user experience, we validated if the multi-agent pipeline reliably supports bidirectional interaction. We evaluated each pipeline component in isolation using correctly-routed inputs, then the end-to-end task success across all inputs to measure user-facing reliability. Two researchers independently evaluated all 152 logged interactions from the user study sessions; inter-rater agreement was substantial (Cohen's $\kappa = 0.81$), with disagreements resolved through discussion. All LLM-based classification in the pipeline uses GPT-4o.

7.1 Oracle Evaluation: Intent Classification

The Oracle classifies each user input into one of three intent categories: *query* for information retrieval from the knowledge base, *contribute* for modification to the knowledge graph, or *expansion* for elaboration on existing nodes. Correct classification is essential because it determines which downstream component, either the Retriever or Map Manager, processes the input.

Data and Method. We assessed all 152 user inputs logged during the user study. Two researchers independently labeled each input, achieving Cohen's $\kappa = 0.84$. We then compared the Oracle's classifications against these ground truth labels.

Results. The Oracle achieved 91.4% overall accuracy. Table 1 shows precision, recall, and F1 scores for each intent category. The primary source of error was confusion between *query* and *expansion* intents (9 cases), which is expected given their semantic overlap: both involve information retrieval, differing primarily in whether new nodes should be generated. Misclassification between *query*/*expansion* and *contribute* was rare (4 cases). The Oracle reliably distinguishes retrieval requests from edit commands.

7.2 Map Manager Evaluation: Edit Execution

The Map Manager executes contribution commands by parsing user intent, planning the required operations, and modifying the knowledge graph accordingly. We evaluate whether the Map Manager correctly executes these commands across multiple dimensions.

Table 1: Oracle Intent Classification Results

Intent Class	Precision	Recall	F1	N
Query	0.89	0.92	0.90	58
Contribute	0.94	0.91	0.92	47
Expansion	0.91	0.87	0.89	47
Overall Accuracy	—	—	**91.4%**	**152**

Data and Method. Of the 47 logged contribution commands, 43 were correctly classified by the Oracle. We examined these correctly-routed commands to isolate Map Manager performance from Oracle errors. Two researchers independently rated each command on four dimensions. 1) *Execution Success* captures whether the system completed without error. 2) *Node Correctness* rates whether the created or modified node contains correct content on a 3-point scale[10]. 3) *Placement Correctness* rates whether the node appears in the correct location in the hierarchy, also on a 3-point scale[11]. 4) *Relationship Correctness* rates whether the edge label accurately describes the relationship on a 3-point scale[12]. Scores were averaged across annotators, with disagreements on execution success resolved through discussion. We also report the *fully correct rate*: the percentage of commands where execution succeeded and all three correctness dimensions received a score of 3.

Results. Table 2 (top section) presents the Map Manager evaluation results. The system achieved 93.0% execution success rate; commands rarely caused system errors. Among successfully executed commands, node correctness was high (M = 2.79/3), while placement and relationship correctness showed more variation (M = 2.58/3 and M = 2.63/3, respectively). The fully correct rate of 78.1% indicates that more than three-quarters of user contributions were integrated exactly as intended. The most common placement errors occurred when users provided ambiguous parent references, such as "add this under climate change" when multiple nodes contained that phrase.

7.3 Adaptive Retriever Evaluation: Retrieval Quality

To validate that the retrieval component provides sufficient answer quality for reliable co-creation, we compared our pipeline against a RAG-only baseline on the same logged queries.

Data and Method. Of the 105 logged queries and expansion requests, 97 were correctly classified by the Oracle. We re-ran these queries through both our pipeline (Raptor hierarchical retrieval with relevance grading) and a naive RAG baseline (standard chunking with vector similarity retrieval). Two researchers rated each response on a 3-point scale[13] (Cohen's κ = 0.76).

[10]Score scale: 3 indicates correct content, 2 indicates minor issues such as truncation, and 1 indicates wrong content.
[11]Score scale: 3 indicates correct parent, 2 indicates reasonable but suboptimal location, and 1 indicates wrong location.
[12]Score scale: 3 indicates appropriate, 2 indicates acceptable but imprecise, and 1 indicates wrong or misleading.
[13]Score scale: 3 indicates correct (fully answers the query), 2 indicates partially correct (addresses the query but incomplete or contains minor errors), and 1 indicates incorrect (fails to answer or provides wrong information).

Table 2: Map Manager, Retriever, and End-to-End Evaluation Results

Component	Metric	Value
Map Manager	Execution Success Rate	93.0%
	Node Correctness (mean)	2.79 / 3
	Placement Correctness (mean)	2.58 / 3
	Relationship Correctness (mean)	2.63 / 3
	Fully Correct Rate	78.1%
	N	43
Retriever	Answer Quality (Naive RAG)	2.34 / 3
	Answer Quality (Our Pipeline)	2.81 / 3
	Correct Rate (Naive RAG)	58.8%
	Correct Rate (Our Pipeline)	82.5%
	N	97
End-to-End	Query / Expansion Success	85.7%
	Contribute Success	80.9%
	Overall Success	83.6%
	N	152

Results. Our pipeline achieved higher answer quality than the naive baseline (2.81/3 vs. 2.34/3; correct rate 82.5% vs. 58.8%; Table 2, middle section), confirming that the hierarchical retrieval design provides sufficient accuracy for the downstream knowledge construction task.

7.4 End-to-End Evaluation: Task Success

The component evaluations above use correctly-routed inputs to isolate each component's performance. We now evaluate end-to-end task success across all inputs to measure user-facing reliability.

Data and Method. We evaluated all 152 user inputs without filtering. Two researchers independently judged whether each task succeeded. For query and expansion tasks, success required that the Oracle correctly classified the input and the generated response was rated at least 2 (partially correct or better). For contribute tasks, success required that the Oracle correctly classified the input, execution succeeded, and placement correctness was at least 2. For failed tasks, researchers attributed the failure to the responsible pipeline component: Oracle misclassification, Map Manager error, or Retriever/response error.

Results. The pipeline achieved 83.6% overall success rate, with query/expansion tasks at 85.7% and contribute tasks at 80.9% (Table 2, bottom section). Among the 25 failed tasks, Oracle misclassification accounted for 52.0% of failures (13 cases), Retriever/response errors for 36.0% (9 cases), and Map Manager errors for 12.0% (3 cases). The relatively higher proportion of Oracle and Retriever errors suggests that intent classification and answer generation are the primary areas for future improvement, while edit execution is comparatively robust.

Summary. These technical results establish that the multi-agent pipeline achieves reliable bidirectional interaction. The Oracle's 91.4% accuracy ensured that user intents were correctly routed to the appropriate pipeline branch. The Map Manager's 78.1% fully-correct rate meant that user contributions were reliably integrated

into the knowledge structure. Our retrieval pipeline outperformed naive RAG on answer quality (2.81 vs. 2.34); the hierarchical design enables effective retrieval across varying levels of granularity. The overall 83.6% end-to-end success rate confirms that participants experienced the intended interaction paradigm.

8 DISCUSSION

Our study compared two designs for knowledge construction: a retrieval-only baseline where users queried documents and received system-generated graph visualizations, and MindTrellis, where users could also shape the evolving knowledge structure through contributions and reorganization. The significant differences across knowledge organization (Q3), depth of exploration (Q6), and slide deck organization (Q11) suggest that allowing users to co-construct the knowledge structure changes how they engage with complex information. Retrieval-only systems such as Sensecape [51] and Graphologue [21] generate visualizations for users to navigate, but the structure remains static. MindTrellis participants actively shaped the representation to reflect their evolving understanding.

Participants' preference for progressive expansion aligns with cognitive load research. Paas [39] found that presenting information gradually helps learners process it more effectively in working memory. Mayer and Moreno [31] identify segmentation as a key principle for reducing extraneous cognitive load. The baseline's simultaneous presentation of all nodes created what P10 described as an "overwhelming experience." In contrast, MindTrellis's incremental expansion allowed participants to "focus on the information they needed." P11 similarly noted that the system "expands the flowchart directly from a parent node, which makes it easier to see the connections."

The value participants placed on contributing to the knowledge structure resonates with research on external cognition and constructive learning. Kirsh [23] argues that creating external representations amplifies cognition by offloading memory, making relationships explicit, and enabling iterative refinement. P1's approach illustrates this pattern: they used MindTrellis to "organize my mind map fully before starting on the slides." The act of structuring knowledge externally supported their subsequent task performance. Educational research quantifies this benefit: Schroeder et al.'s meta-analysis [46] found that students who actively constructed concept maps showed significantly higher learning gains ($g = 0.72$) compared to those who passively studied pre-made maps ($g = 0.43$). These findings further reinforce that contribution, not just retrieval, deepens engagement.

8.1 Situating Findings Relative to Existing Systems

Commercial tools such as NotebookLM and Notion AI already support forms of knowledge-level bidirectionality—users can both query and contribute to underlying knowledge stores. Our study results speak to what additional design choices improve the knowledge construction experience. The significant differences on Q3 (knowledge organization) and Q6 (depth of exploration) emerged from a comparison where the key differentiator was the visual knowledge graph as a shared artifact that users could directly reshape. Participants valued seeing the structure they were building (P8: "allowed me to organize my thoughts as it focused on higher-level ideas") and controlling its organization (P12: "allows me to control the flow of the graph that follows my own mindset"). In NotebookLM, notes and retrieval results exist in separate panes; in Notion AI, the database structure is not visualized as a semantic graph.[14] Our findings suggest that contribution pathways are more effective when contributions are immediately visible within the evolving structure and spatially integrated with retrieved content, rather than residing in a separate view.

Beyond the question of where contributions become visible, our study has implications for the technical challenges that arise when users interact with the knowledge structure through natural language. Systems that achieve bidirectional interaction through representational synchronization [7, 17, 58, 59] partition interaction across structurally explicit boundaries, so the mapping between user action and system response is largely deterministic and the system seldom needs to infer what the user intends. When users both query from and contribute to a shared knowledge structure through natural language, however, the same input channel carries retrieval requests, contribution commands, and ambiguous mixtures of both. Our pipeline evaluation reveals that intent disambiguation is the primary failure mode in this setting: Oracle misclassification accounted for 52% of end-to-end failures (13 of 25 failed tasks). As systems move from synchronizing two views of a single artifact to enabling users and AI to co-construct an evolving knowledge structure through natural language, the disambiguation challenge becomes a central design problem absent from the representational synchronization paradigm.

Our study also reveals how users balance exploration and construction when both are available within a single system. Luminate [50] structures the design space of LLM outputs to support divergent exploration, and Dück et al. [13] support claim retrieval through multiple exploration pathways. Our participants exhibited both exploratory and constructive patterns: some explored broadly before contributing (P1, P4, P5, P10, P12), while others anchored exploration in their own contributions from the start (P3, P6, P8, P11). The coexistence of these patterns suggests that exploration and construction are not sequential phases but interleaved activities that reinforce each other. Luminate's approach of surfacing diverse response dimensions could complement a system like MindTrellis's persistent evolving structure—future systems might integrate structured exploration of LLM outputs as a way to seed or enrich a knowledge graph before and during user contribution.

8.2 Design Implications

We identify three design implications for systems that support human-AI knowledge construction.

First, natural-language co-creation systems need explicit mechanisms for intent disambiguation. In systems that synchronize two views of a single artifact through representational synchronization [7, 17, 58, 59], the editing modality is structurally partitioned and intent interpretation is largely unnecessary. When users instead interact with a shared knowledge structure through natural language, the same input channel carries queries, contribution commands, and ambiguous mixtures of both. P4 noted that

[14] Based on publicly available versions as of early 2026

"sometimes the system didn't quite understand what I was asking for, and I had to rephrase my question." The 52% share of Oracle errors among end-to-end failures confirms that disambiguation is the primary technical bottleneck for this class of system. Deng et al. [10] observe that most language agents "lack interactive mechanisms; when faced with ambiguity, agents confidently commit to an assumed query, leading to incorrect answers." Future systems should provide explicit feedback about how input was interpreted and allow easy correction when misclassification occurs. During our study, participants who encountered misinterpretations typically discovered the mismatch only after the graph had already been modified, requiring manual correction (P4, P9). In systems where the knowledge structure and the input interface coexist as separate representations, such as MindTrellis's graph canvas and chat panel, disambiguation feedback could span both: the chat panel could present the system's interpretation and request clarification, while the graph canvas could preview the intended structural change for the user to confirm or reject before execution. More generally, systems accepting unconstrained natural language input into a shared knowledge structure are likely to face similar challenges [10, 47], and the design of the input interpretation layer deserves as much attention as the knowledge representation itself.

Second, co-creation systems should support flexible information pacing and expansion granularity. Our findings confirm that progressive disclosure reduces cognitive load, but the more consequential design question is what granularity of expansion to offer in a user-editable structure. P8 described how MindTrellis "allowed me to organize my thoughts as it focused on higher-level ideas," suggesting that initial displays should emphasize conceptual structure over detail. However, P9 also noted that "expanding groups together would be a great addition," indicating that users sometimes need to reveal related content simultaneously rather than node-by-node. Sensecape [51] addresses multilevel abstraction, but within a read-only structure. In an editable co-created graph, expansion granularity interacts with the user's ongoing reorganization—expanding a cluster may conflict with manual rearrangements the user has already made. Future systems should support flexible expansion at multiple levels—individual nodes, related clusters, or entire subtrees—and allow users to control the pacing of information revelation as the structure grows.

Third, AI-generated organizational structure should be treated as provisional and user-adjustable at every level. Previous systems for LLM-augmented knowledge exploration [21, 50, 51] generate knowledge structures that users cannot reorganize—the AI's organizational decisions are final. Our study shows that participants actively reshaped the AI's structure across both interaction modalities. P7 and P10 used natural language to reorganize portions of the graph; P10 noted that "typing a command was faster than dragging things around." P9 used direct manipulation, describing how they would "drag the nodes around to organize them the way I wanted." P3 spent significant time "rearranging nodes and adding my own ideas because I wanted the graph to reflect how I think about the topic." Binks et al. [3] found that users adopt diverse organizational strategies when structuring knowledge independently, and no single representational structure is universally effective for all purposes or ways of thinking. The implication extends beyond the placement of individual nodes to the entire topology: the system's choice of hierarchy, grouping logic, and relationship types all encode organizational judgments that may diverge from a given user's mental model. Future co-creation systems should treat AI-generated structure as provisional at every level—placement, grouping, hierarchy, and relationship types—and support user correction through both direct manipulation and natural language. As interaction accumulates, systems could model individual users' organizational preferences to reduce the frequency of corrections over time, a form of computational Theory of Mind [28, 49] applied to knowledge structure rather than dialogue.

8.3 Limitations and Future Work

Our evaluation has limitations that suggest directions for future work. The 12-participant study, while appropriate for formative evaluation of interactive systems [36], involved graduate students with prior AI experience and may not generalize to broader populations. We measured user perceptions but did not conduct independent assessment of task outcomes, such as expert evaluation of slide deck quality. Our pipeline evaluation focused on component-level accuracy rather than long-term system robustness as knowledge graphs grow larger; future work should conduct ablate the multi-agent architecture against simpler alternatives.

Our user study focused on slide deck preparation as a downstream task for knowledge construction. While the task requires synthesizing, organizing, and presenting information from multiple sources, it represents one point in a broader space of knowledge work activities. Preparing for an exam, conducting a literature review, or evaluating a new idea may require different balances of retrieval and contribution. Future work should examine whether the contribution pathway's benefits generalize across these varied task contexts.

Finally, MindTrellis represents knowledge as a hierarchical node-link graph, which is effective for modeling semantic and hierarchical relationships but may not support all organizational strategies equally well. Future work should explore whether alternative visual representations, such as spatial clusters, matrix views, or timelines, might better match certain organizational preferences and complement the node-link graph with representational flexibility.

9 CONCLUSION

In this paper, we present MindTrellis, an interactive visual system to support human-AI collaborative knowledge construction, grounded by established principles in exploratory information seeking and knowledge externalization. MindTrellis enables users to both query document-grounded information and contribute to the knowledge structure by adding concepts, modifying relationships, and reorganizing the hierarchy, producing a co-created knowledge graph where document-derived and user-contributed knowledge coexist. A multi-agent pipeline coordinates intent disambiguation, knowledge placement, and coherence maintenance, with each component validated through quantitative evaluation. A user study with 12 participants compared MindTrellis against a retrieval-only baseline with graph visualization. Overall, participants reported lower cognitive load and enhanced knowledge organization when exploring unfamiliar topics, particularly valuing the ability to progressively

expand the knowledge graph and integrate their own insights into the evolving structure.

ACKNOWLEDGMENTS

This work is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant #RGPIN-2020-03966, the Canada Foundation for Innovation (CFI) John R. Evans Leaders Fund (JELF) #42371, and a gift fund from Adobe.

REFERENCES

[1] Marwan Al-Tawil, Vania Dimitrova, and Dhavalkumar Thakker. 2020. Using knowledge anchors to facilitate user exploration of data graphs. *Semantic Web* 11, 2 (2020), 205–234.

[2] Marwan Al-Tawil, Vania Dimitrova, Dhavalkumar Thakker, and Bilal Abu-Salih. 2023. Emerging Exploration Strategies of Knowledge Graphs. *IEEE Access* 11 (2023), 94713–94731.

[3] Adam Binks, Alice Toniolo, and Miguel A. Nacenta. 2022. Representational transformations: Using maps to write essays. *International Journal of Human-Computer Studies* 165 (2022), 102851. https://doi.org/10.1016/j.ijhcs.2022.102851

[4] Tony Buzan. 2006. *Use your head.* Pearson Education.

[5] Alberto J Cañas, Roger Carff, Greg Hill, Marco Carvalho, Marco Arguedas, Thomas C Eskridge, James Lott, and Rodrigo Carvajal. 2005. Concept maps: Integrating knowledge and information visualization. *Knowledge and information visualization: Searching for synergies* (2005), 205–219.

[6] Guilherme Carneiro, Alice Toniolo, Miguel A. Ncenta, and Aaron J. Quigley. 2021. Text vs. Graphs in Argument Analysis . In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE Computer Society, Los Alamitos, CA, USA, 1–9. https://doi.org/10.1109/VL/HCC51201.2021.9576493

[7] Dan Cașcaval, Mira Shalah, Phillip L. Quinn, Rastislav Bodík, Maneesh Agrawala, and Adriana Schulz. 2021. Differentiable 3D CAD Programs for Bidirectional Editing. *Computer Graphics Forum* 41 (2021). https://api.semanticscholar.org/CorpusID:238259125

[8] Philip R Cohen, Mary Dalrymple, Douglas B Moran, FC Pereira, and Joseph W Sullivan. 1989. Synergistic use of direct manipulation and natural language. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 227–233.

[9] Thomas H Davenport and Laurence Prusak. 1998. *Working knowledge: How organizations manage what they know.* Harvard Business Press.

[10] Mingyi Deng, Lijun Huang, Yani Fan, Jiayi Zhang, Fashen Ren, Jinyi Bai, Fuzhen Yang, Dayi Miao, Zhaoyang Yu, Yifan Wu, Yanfei Zhang, Fengwei Teng, Yingjia Wan, Song Hu, Yude Li, Xin Jin, Conghao Hu, Haoyu Li, Qirui Fu, Tai Zhong, Xinyu Wang, Xiangru Tang, Nan Tang, Chenglin Wu, and Yuyu Luo. 2025. InteractComp: Evaluating Search Agents With Ambiguous Queries. arXiv:2510.24668 [cs.CL] https://arxiv.org/abs/2510.24668

[11] Reinildes Dias. 2011. Concept maps powered by computer software: a strategy for enhancing reading comprehension in English for Specific Purposes. *Revista Brasileira de Linguística Aplicada* 11 (2011), 896–911.

[12] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 601–610. https://doi.org/10.1145/2623330.2623623

[13] Moritz Dück, Steffen Holter, Robin Shing Moon Chan, Rita Sevastjanova, and Mennatallah El-Assady. 2025. Finding Needles in Document Haystacks: Augmenting Serendipitous Claim Retrieval Workflows. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1003, 17 pages. https://doi.org/10.1145/3706598.3713715

[14] Martin J Eppler and Remo A Burkhard. 2008. Knowledge visualization. In *Knowledge management: concepts, methodologies, tools, and applications*. IGI Global, 781–793.

[15] Aliye Erdem. 2017. Mind Maps as a Lifelong Learning Tool. *Universal Journal of Educational Research* 5, n12A (2017), 1–7.

[16] J. Nathan Foster, Michael B. Greenwald, Jonathan T. Moore, Benjamin C. Pierce, and Alan Schmitt. 2007. Combinators for bidirectional tree transformations: A linguistic approach to the view-update problem. *ACM Trans. Program. Lang. Syst.* 29, 3 (May 2007), 17–es. https://doi.org/10.1145/1232420.1232424

[17] Brian Hempel, Justin Lubin, and Ravi Chugh. 2019. Sketch-n-Sketch: Output-Directed Programming for SVG. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 281–292. https://doi.org/10.1145/3332165.3347925

[18] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)* 54, 4 (2021), 1–37.

[19] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human–computer interaction* 1, 4 (1985), 311–338.

[20] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems* 33, 2 (2022), 494–514.

[21] Peiling Jiang, Jude Rayan, Steven P. Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, Article 3, 20 pages. https://doi.org/10.1145/3586183.3606737

[22] Rebecca Oluwayimika Kasumu and Rebecca Oluwayimika. 2022. CONCEPT MAPPING AS A TEACHING STRATEGY: BENEFITS AND CHALLENGES IN HIGHER INSTITUTION. *International Journal Of Trendy Research In Engineering And Technology* 6, 06 (2022), 5–10.

[23] David Kirsh. 2010. Thinking with external representations. *AI & SOCIETY* 25 (2010), 441–454. https://api.semanticscholar.org/CorpusID:16683273

[24] LangChain. 2024. LangGraph: A LangChain Application. https://www.langchain.com/langgraph Accessed: 2024-10-08.

[25] Jill H. Larkin and Herbert A. Simon. 1987. Why a Diagram is (Sometimes) Worth Ten Thousand Words. *Cognitive Science* 11, 1 (1987), 65–100. https://doi.org/10.1016/S0364-0213(87)80026-5

[26] Jae Hwa Lee and Aviv Segev. 2012. Knowledge maps for e-learning. *Computers & Education* 59, 2 (2012), 353–364.

[27] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401 [cs.CL] https://arxiv.org/abs/2005.11401

[28] Huao Li, Yu Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. 2023. Theory of Mind for Multi-Agent Collaboration via Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 180–192. https://doi.org/10.18653/v1/2023.emnlp-main.13

[29] Michael Xieyang Liu, Tongshuang Wu, Tianying Chen, Franklin Mingzhe Li, Aniket Kittur, and Brad A Myers. 2024. Selenite: Scaffolding Online Sensemaking with Comprehensive Overviews Elicited from Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 837, 26 pages. https://doi.org/10.1145/3613904.3642149

[30] Cristiane Tolentino Machado and Ana Amélia Carvalho. 2020. Concept mapping: Benefits and challenges in higher education. *The Journal of Continuing Higher Education* 68, 1 (2020), 38–53.

[31] Richard E. Mayer and Roxana Moreno. 2003. Nine Ways to Reduce Cognitive Load in Multimedia Learning. *Educational Psychologist* 38 (2003), 43 – 52. https://api.semanticscholar.org/CorpusID:13667935

[32] Robert Meyer. 2010. Knowledge visualization. *Trends in information visualization* 23 (2010), 23–30.

[33] Sheryl L Miller and J Gregory Trafton. 1999. Improving information search using natural language and direct manipulation tools in a multimodal interface. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 43. SAGE Publications Sage CA: Los Angeles, CA, 462–466.

[34] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. 2018. Never-ending learning. *Commun. ACM* 61, 5 (April 2018), 103–115. https://doi.org/10.1145/3191513

[35] Martin Nečaský and Štěpán Stenchlák. 2022. Interactive and iterative visual exploration of knowledge graphs based on shareable and reusable visual configurations. *Journal of Web Semantics* 73 (2022), 100713.

[36] Jakob Nielsen. 1993. Usability engineering. In *The Computer Science and Engineering Handbook*. https://api.semanticscholar.org/CorpusID:260423343

[37] JD Novak. 1984. Learning how to learn. *Press Syndicate of the University of Cambridge* (1984).

[38] Angela M O'donnell, Donald F Dansereau, and Richard H Hall. 2002. Knowledge maps as scaffolds for cognitive processing. *Educational psychology review* 14 (2002), 71–86.

[39] Fred Paas. 1992. Training strategies for attaining transfer of problem-solving skill in statistics: A cognitive-load approach. *Journal of Educational Psychology* 84 (1992), 429–434. https://api.semanticscholar.org/CorpusID:145052264

[40] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of the International Conference on Intelligence Analysis*, Vol. 5. McLean, VA, USA, 2–4.

[41] Ehsan Rassaei. 2019. Effects of two forms of concept mapping on L2 reading comprehension and strategy awareness. *Applied Linguistics Review* 10, 2 (2019), 93–116.

[42] Daniel M. Russell, Mark J. Stefik, Peter Pirolli, and Stuart K. Card. 1993. The cost structure of sensemaking. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (Amsterdam, The Netherlands) *(CHI '93)*. Association for Computing Machinery, New York, NY, USA, 269–276. https://doi.org/10.1145/169059.169209

[43] Ammar H Safar, Yaqoub J Jafer, and Mohammad A Alqadiri. 2014. Mind maps as facilitative tools in science education. *College Student Journal* 48, 4 (2014), 629–647.

[44] Bahareh Sarrafzadeh, Alexandra Vtyurina, Edward Lank, and Olga Vechtomova. 2016. Knowledge graphs versus hierarchies: An analysis of user behaviours and perspectives in information seeking. In *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval*. 91–100.

[45] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. arXiv:2401.18059 [cs.CL] https://arxiv.org/abs/2401.18059

[46] Noah L. Schroeder, John C. Nesbit, Carlos J. Anguiano, and Olusola O. Adesope. 2018. Studying and Constructing Concept Maps: a Meta-Analysis. *Educational Psychology Review* 30, 2 (June 2018), 431–455. https://doi.org/10.1007/s10648-017-9403-9

[47] Reza Shahriari, Eric D. Ragan, and Jaime Ruiz. 2025. Natural Language Interaction for Editing Visual Knowledge Graphs. In *Proceedings of the 13th Knowledge Capture Conference 2025 (K-CAP '25)*. Association for Computing Machinery, New York, NY, USA, 26–34. https://doi.org/10.1145/3731443.3771344

[48] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval Augmentation Reduces Hallucination in Conversation. arXiv:2104.07567 [cs.CL] https://arxiv.org/abs/2104.07567

[49] Winnie Street. 2024. LLM Theory of Mind and Alignment: Opportunities and Risks. arXiv:2405.08154 [cs.HC] https://arxiv.org/abs/2405.08154

[50] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/3613904.3642400

[51] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–18. https://doi.org/10.1145/3586183.3606756

[52] Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. 2024. Cognitive Architectures for Language Agents. *Transactions on Machine Learning Research* (2024).

[53] Lei Wang, Chen Ma, Xueyang Feng, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (2024), 186345.

[54] Minhong Wang and Michael J Jacobson. 2011. Guest editorial-knowledge visualization for learning and knowledge management. *Journal of Educational Technology & Society* 14, 3 (2011), 1–3.

[55] Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. Executable Code Actions Elicit Better LLM Agents. arXiv:2402.01030 [cs.CL] https://arxiv.org/abs/2402.01030

[56] Amila Wickramasinghe, Nimali Widanapathirana, Osuka Kuruppu, Isurujith Liyanage, and IMK Karunathilake. 2011. Effectiveness of mind maps as a learning tool for medical students. *South East Asian J Med Educ* 1 (01 2011).

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

[58] Yifan Wu, Joseph M. Hellerstein, and Arvind Satyanarayan. 2020. B2: Bridging Code and Interactive Visualization in Computational Notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 152–165. https://doi.org/10.1145/3379337.3415851

[59] Katherine Ye, Wode Ni, Max Krieger, Dor Ma'ayan, Jenna Wise, Jonathan Aldrich, Joshua Sunshine, and Keenan Crane. 2020. Penrose: from mathematical notation to beautiful diagrams. *ACM Trans. Graph.* 39, 4, Article 144 (Aug. 2020), 16 pages. https://doi.org/10.1145/3386569.3392375

A PROMPTS

A.1 Parse Query

```
You are a helpful assistant that classifies the
    user's query and generates an optimized query
    content.
When there's ambiguity in the user's query, use
    the chat history to infer the user's intention
    .
Ensure the query content is clear and can be
    executed without seeing the chat history.

There are three types of user's query:
1. Information Retrieval Queries:
   - Examples: "What is the definition of X?", "Is
       X the best player in the world?"
   - Generate query category: "search"
   - Generate query content: Preserve the original
       query without modifying meaning
2. Graph Editing Commands:
   - Examples: "Add a new concept called 'X' under
       the concept 'Y'"
   - Generate query category: "edit"
   - Generate query content: Preserve the original
       command without modification
3. Expansion Requests:
   - Examples: "Tell me more about X", "Explain X
      in more detail", "Elaborate on X"
   - Generate query category: "expansion"
   - Generate query content: "What are the sub-
      topics covered in the document related to '
      X'?"

User: {query}
Context: {chat_history}
```

The Oracle Module uses this classification system to route queries to the appropriate processing pipeline and to reformat ambiguous queries for optimal processing.

A.2 Update Graph

```
User's question: {query}
Answer: {response}

Update the graph based on the question and answer:

1. Create a new node for the answer if the answer
    does not fit under any existing node.
   Link the new node to the existing nodes that
       are related to the answer.
   Consider both the node name and description
       when identifying related existing nodes.
   For example, if the user asks "What is the
       definition of XXX?", check if there's a
       node
   with the name "XXX" or a description containing
        "XXX".

2. Break down the user's question and the answer
    into key points.

3. Maintain hierarchy: general key points as
    parent nodes, specific details as child nodes.

4. For each key point:
   a. Identify the relevant nodes in the graph
       related to the key point.
   b. If the key point is a sub-topic of an
       existing node, add it as a child node.
   c. If the key point is a parent-topic of an
       existing node, add it as a parent node.

Remember: You're creating a hierarchical knowledge
    graph, not a flat list.
```

This prompt guides the system in maintaining hierarchical relationships when updating the knowledge graph based on new information from user interactions.

A.3 Query Graph

```
You are an assistant for question-answering tasks.
    You will be given a question and a context.
Use ONLY the following pieces of retrieved context
    to answer the question.
If you lack sufficient information from the
    context, respond with 'I don't know'.
Do not fabricate or assume any information not
    present in the context.

Your answer should resemble a hierarchical map,
    describing the relationships between each
    topic and the central keyword of the user's
    input.
For each topic, explain how it is related to the
    central keyword, using specific information
    from the context.

The context is: {context}
```

This prompt instructs the system to generate structured responses resembling a knowledge graph based strictly on the provided context, maintaining clear relationships between topics and the central concept.

A.4 Refine Query

```
You are an intelligent query refiner. Your task is
    to analyze the user's original query and the
    response from the graph retriever, then
    generate a refined query for the RAG retriever
    .

Guidelines:
1. Focus on the parts of the query that cannot be
    answered by the graph.
2. Make the refined query more specific and
    targeted.
3. Remove any parts of the query that can already
    be answered by the graph.
4. Ensure the refined query is clear and self-
    contained.
```

```
5. If the entire query can be answered by the
    graph, generate a minimal query to confirm or
    expand on the information.
Directly output the refined query, do not output
    any other text.

Original query: {original_query}
Graph retriever response: {graph_response}
```

This prompt helps the system refine user queries by identifying information gaps in the current knowledge graph, ensuring that subsequent retrievals are targeted and non-redundant.

A.5 Query Knowledge Base

```
You are an assistant for question-answering tasks.
     You will be given a question and a context.
Use ONLY the following pieces of retrieved context
     to answer the question.
If you lack sufficient information from the
    context, respond with 'I don't know'.
Do not fabricate or assume any information not
    present in the context.
Your answer should resemble a mind map, describing
     the relationships between each topic and the
    central keyword of the user's input.
For each topic, explain how it is related to the
    central keyword, using specific information
    from the context.

The context is: {context}
```

This prompt guides the system to generate structured responses organized as a knowledge graph that maintain clear relationships.

A.6 Generate Suggestions

```
You are an intelligent agent responsible for
    generating suggestions for expanding a
    knowledge graph. Your task is to determine the
     most logical relationships between potential
    new content and one specific existing node.

1. Read the given existing node content carefully
    and understand the context.
2. Based on your knowledge and the context of the
    existing node, generate relevant suggestions
    for expansion.
3. Review the existing suggestions of the current
    node and do not suggest the same topic twice.
4. Determine the most appropriate relationship
    between the new content and the existing node.
5. Provide a list of suggestions, where each
    suggestion includes:
   - A topic that could be added as a child of the
       current node
   - A brief description of that topic as a full,
       informative sentence
   - The relationship between the new content and
       the existing node

Ensure that:
1. The suggestions are directly related to the
    existing node with a logical relationship.
2. Only include content that directly fits as
    children of the current node.
3. If you don't have any suggestions, just return
    an empty list.
4. Aim to provide 3-5 relevant and diverse
    suggestions for expanding the graph, DO NOT
    EXCEED 5.
5. The description should be a complete,
    informative sentence that addresses specific
    aspects or examples related to the topic.

Example of a good description:
Topic: 'Applications of Machine Ethics'
Description: 'Machine ethics is applied in various
     real-world scenarios, including autonomous
    vehicles making moral decisions in potential
    accident situations, AI systems in healthcare
    prioritizing patient care, and military AI
    navigating complex ethical dilemmas in combat
    situations.'

Existing nodes on the graph: {existing_nodes}
Expanding from node: {node_info}
Existing suggestions: {existing_suggestions}
Related content in the docs: {content}
```

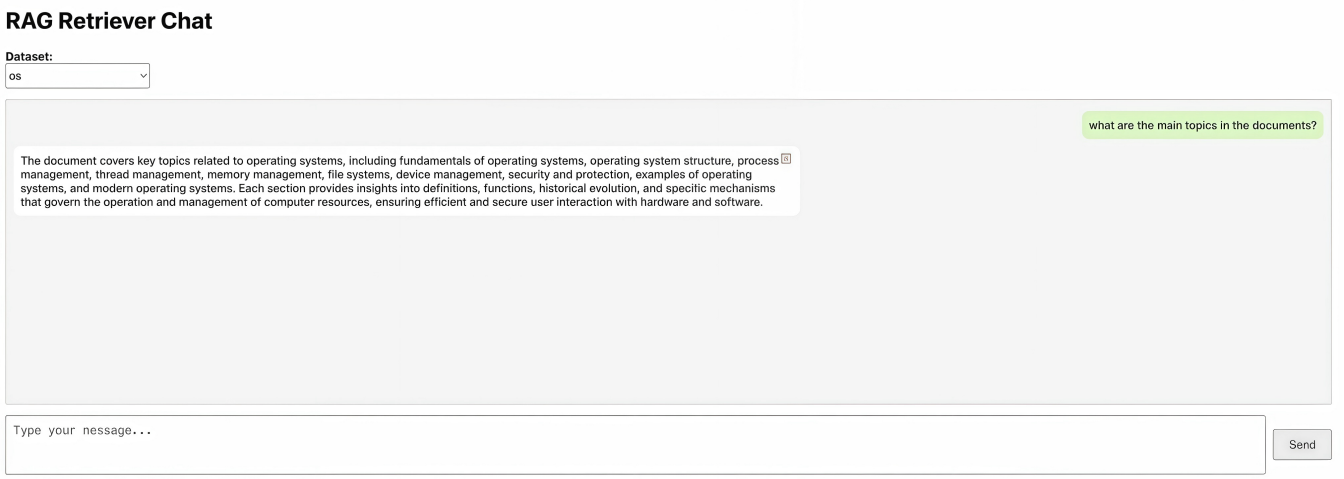
This prompt guides the system in generating contextually relevant suggestions for expanding specific nodes in the knowledge graph, ensuring diversity and logical relationships.

A.7 Rewrite Final Output

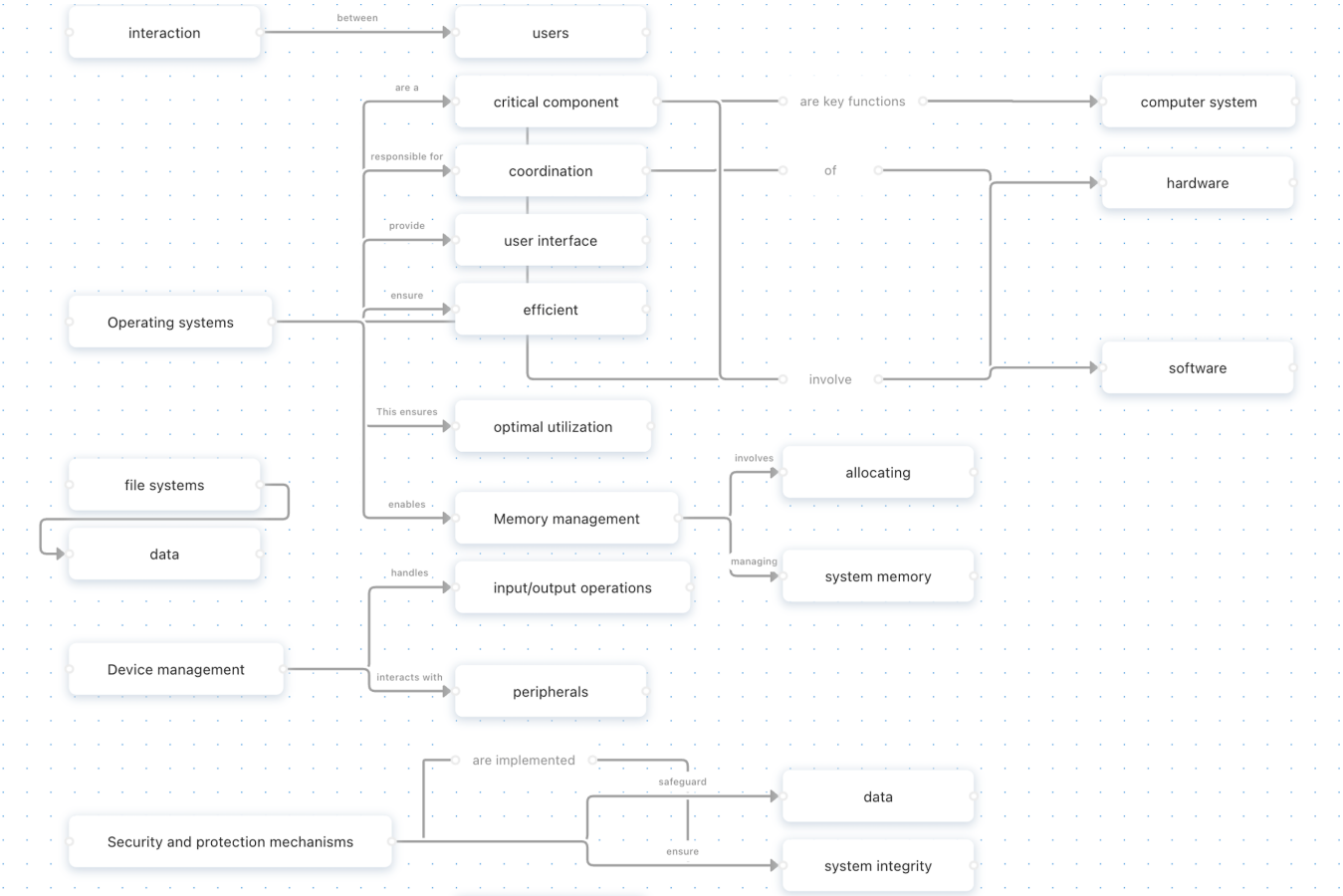
```
User's input: {input}
Refined query based on the input: {query}

Your task is to rewrite the following answer: {
    answer} such that the response answers both
    the user's input and the refined query.
After the final answer, add a reference section
    that lists the sources of the answer: {sources
    }
After the reference section, add a note to the
    user that the question is asked specifically
    on the node {node_name}

Rewriting guidelines:
1. Improve readability without modifying the
    content of the original answer.
2. The final answer should answer the user's input
     and the refined query.
3. Keep the answer concise and to the point.
4. Improve formatting for clarity if possible.
5. The final answer should introduce the main
    themes directly. Do not have text "High-level
    summary" or "Detailed bullet points" in the
    final answer.
```

This prompt instructs the system to produce concise, well-formatted responses that address user queries.

B BASELINE USER INTERFACE



(a) Baseline RAG Retriever Interface. Participants select a dataset from the dropdown menu and pose questions to receive relevant responses.



(b) Baseline Visualization Interface. Participants input responses from the retriever and explore the knowledge base through automatically generated node-link diagrams.

Figure 7: Baseline system interfaces used in the user study.