A Recommendation System for Ideation: Enhancing Supermind Ideator

by

Daniel Papacica

S.B. Computer Science and Engineering, Massachusetts Institute of Technology, 2023

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by: Daniel Papacica
Department of Electrical Engineering and Computer Science
May 17, 2024

Certified by: Thomas W. Malone
Patrick J McGovern Professor of Management, Thesis Supervisor

Accepted by: Katrina LaCurts
Chair
Master of Engineering Thesis Committee

A Recommendation System for Ideation: Enhancing Supermind Ideator

by

Daniel Papacica

Submitted to the Department of Electrical Engineering and Computer Science
on May 17, 2024 in partial fulfillment of the requirements for the degree of

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

ABSTRACT

Recommendation systems are widely utilized across various domains such as e-commerce, entertainment, and social media to enhance user experience by personalizing content and suggestions. Despite their widespread use, these systems are rarely applied to the ideation process, presenting unique challenges due to the inherently creative and complex nature of generating and developing novel ideas. This thesis details the creation and assessment of a recommendation system for the Supermind Ideator platform, aimed at enhancing the creative ideation processes. The recommendation system leverages machine learning techniques to dynamically adapt to user input statements based on statement "scope", a sub-task that is thoroughly explored and tested in this paper. "Scope" is then integrated into the recommendation system's static rules-based algorithm to suggest the next best Supermind Design "move". This work not only contributes a practical tool to the field of ideation but also extends the theoretical understanding of recommendation systems in facilitating complex, subjective cognitive tasks.

Thesis supervisor: Thomas W. Malone
Title: Patrick J McGovern Professor of Management

Acknowledgments

I would like to thank Steven Rick for his continuous support throughout the entire research process. From brainstorming approaches for this recommendation system to analyzing how to identify and define "scope" for ideation tasks, Steven has helped me with many different tasks. Above all else, Steven helped me understand how to manage my approaches to solving research problems and not let them get too obscure. He was consistently available when I had questions, general or specific, and made time to understand my thought process behind different tasks.

I would like to thank Jennifer Heyman for understanding the problems I was trying to explore, and thoroughly assisting me in the experimentation process. Jennifer's support with setting up experiments and surveys provided me with incredibly useful insight on the tasks I was working on.

I would like to thank my advisor, Prof. Thomas W. Malone, for giving me an incredible opportunity to conduct research for MIT's Center for Collective Intelligence and for guiding important brainstorming sessions that pushed my understanding of ideation and Supermind Design.

I would like to thank my family. Sports and academics always went hand in hand, and I'm glad they always pushed me to pursue both to the fullest extent. I would not be where I am today without their support.

Last, but not least, I would like to thank the whole team at CCI. I thoroughly enjoyed the weekly discussions with the team to understand the latest trends in Generative AI and how to apply my knowledge to ideation and Supermind Design. These discussions were ever so insightful and helpful throughout this project, and also helped me feel at home with this lab.

Contents

Title page 1

Abstract 3

Acknowledgments 5

List of Figures 9

List of Tables 11

1 Introduction 13
- 1.1 Ideator Platform 13
  - 1.1.1 Supermind Design Methodology 14
- 1.2 Importance of Ideation Recommendations 15
- 1.3 Thesis Structure Overview 16

2 Background and Related Work: 17
- 2.1 Previous Ideator Work/Research 17
- 2.2 Data Analysis of Current Ideator Platform Usage 18
  - 2.2.1 User Engagement Depth 18
  - 2.2.2 Single Session Usage 18
  - 2.2.3 Session Length and User Engagement 19
  - 2.2.4 Exploring vs On-topic Sessions 19
  - 2.2.5 User Preferences and Personalization 20
  - 2.2.6 Gaps and Opportunities 21
- 2.3 Previous Recommendation Systems for Ideation 21

3 Recommendation System Design Methodology 23
- 3.1 Research Objectives, Workflow, and Data 23
- 3.2 Version 1 of Recommendation System (static rules-based) 23
  - 3.2.1 Recommendation System Overview 24
- 3.3 Incorporating Problem “Scope” 27
  - 3.3.1 Intuition Behind Unsupervised ML Approach 28
  - 3.3.2 Implementing Approach/Experimentation 28
- 3.4 Analysis of Human Feedback/Labeling 34

4 Experiment, Evaluation Discussion **39**
4.1 Experiment Structure . . . 39
4.2 Evaluation . . . 39
4.2.1 Assessment Criteria . . . 39
4.2.2 Human Evaluation Results . . . 41
4.3 Discussion . . . 43
4.3.1 Recommendation System Strengths . . . 43
4.3.2 Recommendation System Weaknesses . . . 43
4.3.3 Potential Further Experiments . . . 44

5 Conclusions and Further Work **47**
5.1 Ideator Platform Integration of Recommendation System . . . 47
5.2 Ideation Recommendation System Advantages . . . 48
5.3 Future Works . . . 48
5.3.1 Dynamic RNN-based Recommendation System . . . 48
5.3.2 Variational Autoencoder for "Idea" Encoding . . . 49

A Feature Descriptions and Rule Set **51**
A.1 Feature Descriptions . . . 51
A.2 Explanation of Rule Set . . . 52

B Code Listings **55**
B.1 Ideator API Integration . . . 55
B.2 Scope Analysis Inference . . . 56

References **63**

List of Figures

2.1 Graph illustrating difference in Exploring vs. On-Topic sessions 20

3.1 Illustration of Double Diamond process. 24
3.2 Overview of Recommendation System Rule Set 25
3.3 Visualization of Main Topic Clusters 32
3.4 Visualization of Sub-Topic (Refined) Clusters for Largest Main Cluster . . . 32
3.5 Demonstration of how users were tasked with labeling problem statements. . 35
3.6 Distribution of 'General' and 'Specific' human labels. 35
3.7 Accuracy of human labels when compared to system produced labels. 36
3.8 Direct human label accuracy when there was a distinct consensus label. . . . 36

4.1 Example of Idea Sequence. 40
4.2 Distribution of ratings on entire sequence based on criteria. 42
4.3 Distribution of idea selection in sequence based on criteria. 42

5.1 Integration of recommendation system in Ideator Platform. 47

List of Tables

2.1 Proportion of Session Types Based on User Data 20
2.2 User Preferences Overview 20

3.1 Optimization Results Comparison 37

4.1 Human Ratings Overview 41

Chapter 1

Introduction

Recommendation systems are known to encourage and enhance user interaction within various platforms, ideally for the benefit of their experience and results. Although recommendation systems can typically be integrated into commercial platforms (e.g., e-commerce, social media, travel, etc.) smoothly, there arises an issue when the process requiring recommendation generation is complex or has ambiguous structure. This particular issue appears for the ideation process since it is an inherently personal process that can be carried out by following completely different types of frameworks or none at all.

Despite their potential to foster creativity and more meaningful sessions, recommendation systems face challenges like sparsity, scalability issues, and aligning with diverse user preferences (also known as the "gray sheep" problem). These complexities are particularly pronounced in the ideation process.

To address this, we can leverage a fixed framework–the Supermind Design Methodology–in conjunction with a machine learning-based recommendation system. This approach aims to effectively utilize machine learning methods and static-rules based algorithms for recommendation within Supermind Ideator (a generative AI based platform for ideation), enhancing user engagement and promoting deeper, more innovative processes. The work illustrated in this proposal seeks to experiment using a machine learning and static-rules based approach to tackle the problem of supporting recommendation generation for users going through the process of ideation. Our preliminary analysis suggests that such a recommendation system could enhance user capabilities during ideation and encourage more thoughtful and engaging ideation sessions.

1.1 Ideator platform

The Supermind Ideator is a digital tool specifically crafted to facilitate the ideation process, aiding users in generating and refining ideas for solving complex problems. At its core, the platform is structured to guide users into a focused state of idea generation and reflection, ensuring an optimal environment for creative thinking and problem-solving.

Key Features of the Supermind Ideator Platform:

The platform provides users with three main options—Explore Problem, Explore Solutions, and More Choices. These "move sets" act as a supportive thinking framework, guiding users through the process of problem definition and solution generation.

Explore Problem: This move set utilizes the basic design moves and experimental moves from the Supermind Design Methodology to support the problem definition phase. It helps users in generalizing and specializing the various aspects of their problem, drawing relevant analogies, and identifying potential gaps in their problem statement.

Explore Solutions: This move set focuses on the solution generation phase, incorporating the Supermind Design moves from the Supermind Design methodology. It assists users in considering how different groups (such as markets, communities, and democracies) and innovative cognitive processes or technologies could be employed to address their problem.

More Choices: This option allows users to select individual moves according to their needs. It also introduces a creative parameter known as "Creativity," which corresponds to the GPT API's "temperature" setting. This parameter offers three levels—Low, Medium, and High—equating to temperatures of 0.7, 1.0, and 1.3, respectively which control the randomness of outputs.

The platform also enables users to interact with and provide feedback on the generated ideas. Each idea can be rated with a Thumbs Up or Thumbs Down button, and users can Bookmark ideas they find particularly valuable, allowing for future reference.

The Supermind Ideator, powered by OpenAI's Generative Pre-trained Transformer (chatGPT), leverages the ability of AI to summarize vast amounts of information, providing users with an array of useful insights. This, combined with the Supermind Design Methodology, creates a comprehensive and user-centric tool that significantly enhances the creative problem-solving process.

1.1.1 Supermind Design Methodology

The Supermind Design Methodology was specifically created to aid users through the ideation process and help people achieve creative solutions to important problems. By employing a set of conceptual moves, the methodology spurs creativity in designing collectively intelligent groups or "superminds" while also assisting individuals through the ideation process.

The Supermind Design Framework comprises three types of moves when a particular person or group is ideating on a problem:

Basic Design Moves:
Zoom In - Parts: Identify the parts of the problem.
Zoom In - Types: Delineate the types of the problem.
Zoom Out - Parts: Understand what the problem is a part of.
Zoom Out - Types: Determine what the problem is a type of.

Analogize: Explore analogies for the problem.

Supermind Design Moves:
Groupify: Identify types of groups that could help solve the problem, such as democracies, markets, or communities.
Cognify: Determine group cognitive processes that could aid in solving the problem, including creating, deciding, sensing, remembering, and learning.
Technify: Evaluate how technologies could be utilized to address the problem.

Experimental Moves:
Reflect: Identify what is missing from the current problem statement.
Reformulate: Explore how the problem could be reformulated.
Case examples: Relate the problem to real-world examples of companies and products.

The Supermind Ideator integrates this methodology to optimize the ideation process. As mentioned in the previous section, it leverages the conceptual moves from the Supermind Design Framework as part of its "move sets" in the Explore Problem and Explore Solutions options, as well as within the More Choices feature. By doing so, it provides users with a structured yet flexible approach to creative problem-solving, thereby facilitating the generation of innovative solutions to complex problems.

1.2 Importance of Ideation Recommendations

The proposed recommendation system will play a crucial role in aiding users to effectively expand their problem space and systematically converge towards feasible solutions. By providing tailored suggestions, the system ensures that users can explore a wide array of perspectives and approaches, fostering a comprehensive understanding of the problem at hand before arriving at a solution.

In the context of the Supermind Ideator platform, the recommendation system will enhance the user experience and improve their ideation process through better design methodology facilitation and exploration. This is achieved by offering contextual and relevant suggestions, which are grounded in the Supermind Design Methodology. As users navigate through the "move sets", the recommendation system actively works to present options that are not only aligned with the users' inputs but also challenge their thinking by introducing novel perspectives and approaches.

Moreover, the Ideator platform's integration of the Supermind Design Framework ensures that the recommendations are not just arbitrary but are systematically derived from a set of conceptual moves specifically designed to spur creativity and facilitate problem-solving. This will make the recommendations highly effective in guiding the user through the ideation process, ultimately leading to more innovative and structured solutions.

In summary, the recommendation system will be a vital component of the Ideator platform, actively contributing to a richer and more productive user experience. By providing personalized, contextual, and methodology-driven suggestions, the system empowers users to explore the full potential of their creative capacities and systematically approach complex problems.

1.3 Thesis Structure Overview

- **Chapter 2** provides background on the Ideator platform where the recommendation system will be used and contextualizes the system through analyzing previous recommendation systems/frameworks for ideation.
- **Chapter 3** illustrates the design and methodology behind the recommendation system
- **Chapter 4** details the experimentation and evaluation of the recommendation system so far.
- **Chapter 5** concludes the initial innovation of this work and potential future paths of enhancement.
- **Appendix A** lists feature descriptions and rule sets used in the recommendation system.
- **Appendix B** provides informative code snippets of how the recommendation system was implemented in the Ideator platform.

Chapter 2

Background and Related Work:

2.1 Previous Ideator Work/Research

The Ideator platform's evolution and its role in the realm of collective intelligence have been marked by significant developments and insights from past research.

Initially, creative problem-solving support involved techniques like brainstorming and design thinking, along with software tools for idea recording and sharing. The advent of generative AI technologies has considerably enhanced this domain. These AI systems can produce a wide range of ideas, some of which might never have occurred to human users, thus expanding the creative possibilities. This capability is particularly beneficial when used to augment human creativity, helping users to select or derive inspiration from AI-generated ideas.

The MIT Center for Collective Intelligence emphasizes the synergy between humans and computers to achieve collective intelligence surpassing individual capabilities. Various projects have been published under this initiative, including:

1. Supermind Ideator: Exploring generative AI to support creative problem-solving (2023): This paper focuses on enhancing creative problem-solving particularly through a unique system that integrates large language models and a user-friendly interface to generate diverse ideas and solutions. This system, using the Supermind Design Methodology, not only aids in brainstorming and ideation but also provides a structured approach to navigating complex problem spaces. This project exemplifies the CCI's commitment to harnessing the synergy between human intelligence and advanced AI technologies to tackle creative challenges. [1]

2. DesignAID: Using Generative AI and Semantic Diversity for Design Inspiration (2023): This project showcases an advanced generative AI tool designed to augment the early stages of creative design. It employs large language models and image generation software to broaden the scope of idea exploration, offering a diverse range of verbal and visual concepts. This tool demonstrates a significant leap in facilitating and inspiring the creative process, directly contributing to the Ideator platform's capability to enhance creative problem-solving and ideation. [2]

3. A Test for Evaluating Performance in Human-Computer Systems (June 2022): This work underscores the evolving role of AI in enhancing human-computer interaction, a concept central to the Ideator platform's development, where generative AI technologies like GPT play a significant role in idea generation and creative problem-solving. [3]

4. Supermind Design for Responding to Covid-19 (February 2022): This paper highlights the application of the Supermind Design methodology in addressing contemporary global challenges, akin to the Ideator platform's approach in leveraging collective intelligence for problem-solving. [4]

5. How American Adults Obtain Work Skills: Results of a New National Survey (2021): This study examines the dynamics of skill acquisition and training among American adults. It provides insights into the distribution and nature of employer-provided training, highlighting disparities in access based on race, ethnicity, and educational attainment. The findings from this survey underscore the importance of continuous learning and skill development in the modern workforce, a theme that aligns with the Ideator platform's goal of integrating AI for creative problem-solving and idea generation. [5]

These publications reflect the ongoing research and development in collective intelligence at MIT CCI and neighboring labs, directly influencing the Ideator platform's approach to utilizing AI and collective human intelligence for innovative problem-solving.

Overall, the Ideator platform represents a significant advancement in leveraging collective intelligence for creative problem-solving. While it has shown substantial potential in assisting users, ongoing research and development have been aimed at fully realizing this potential, with evaluations of user experience, the speed and quality of idea generation, and other factors.

2.2 Data Analysis of Current Ideator Platform Usage

This section presents an analytical overview of the current usage patterns of the Ideator platform, leveraging statistical data to identify key trends and areas for improvement.

2.2.1 User Engagement Depth

The analysis begins by examining the extent to which users engage in detailed ideation processes beyond basic functionalities such as "Explore Problem" or "Explore Solution". The data indicates potential for increasing depth in user engagement.

2.2.2 Single Session Usage

A significant portion of users, accounting for 62.22%, engage in only a single session, indicating a potential drop-off in engagement or a lack of incentive to continue the ideation process beyond the initial stage.

2.2.3 Session Length and User Engagement

Further analysis focuses on session lengths, providing insights into the depth of user engagement:

- Maximum sessions by a single user: 13 sessions
- Average sessions per user: 1.91 sessions
- Maximum 'problem_queries' in a single session: 70 queries
- Average 'problem_queries' per session: 12.83 queries
- Maximum unique 'problem_queries' in a session: 8 queries
- Average unique 'problem_queries' per session: 1.47 queries

This data suggests that while some users engage deeply, most tend to have shorter and less varied sessions. The previous information highlights the need to incorporate a way to encourage/guide users to continue their ideation process which can be achieved through a recommendation system.

2.2.4 Exploring vs On-topic Sessions

In this section, we categorize user sessions on the Ideator platform into two distinct types: 'exploring' and 'on-topic'. This categorization is based on the analysis of the content of 'problem_queries' within individual sessions. The differentiation between these two session types is crucial for understanding user behavior and preferences in the context of ideation.

Methodology for Categorization:
Calculation Basis: The distinction between 'exploring' and 'on-topic' sessions is calculated using the cosine similarity measure between 'problem_queries' within a session.
Cosine Similarity: This metric measures the cosine of the angle between two vectors in a multi-dimensional space, in this case, the vectors representing 'problem_queries'. A higher cosine similarity implies greater similarity between queries.
Threshold Setting: A threshold value for cosine similarity is established to differentiate between exploring and on-topic sessions. Sessions with a mean cosine similarity below the threshold are classified as 'exploring', indicating a greater diversity in the content of 'problem_queries'. Conversely, sessions with a mean cosine similarity above the threshold are classified as 'on-topic', suggesting a more consistent theme or topic throughout the session. The threshold of 0.7 is chosen based on empirical observations within the specific problem query dataset and reflects a balance that is found to effectively distinguish between different types of interactions in the data. This provides a practical tool for categorization in the absence of a universal standard for such thresholds.

The statistics of table 2.1 suggest that a majority of the sessions maintain a consistent thematic focus. However, a significant portion of sessions exhibit exploratory behavior, with users exploring a wider range of ideas within a single session. The method of calculating

Table 2.1: Proportion of Session Types Based on User Data

Session Type	Proportion (%)
On-topic (Focused) Sessions	75.66
Exploratory Sessions	24.34

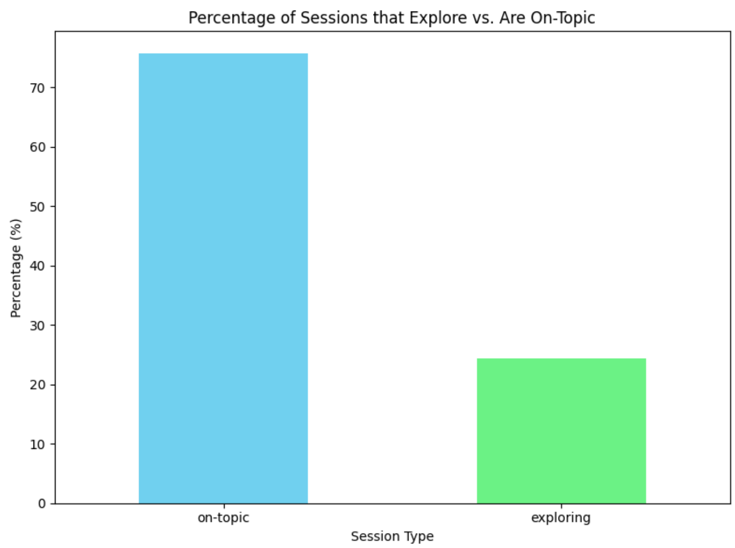

Figure 2.1: Graph illustrating difference in Exploring vs. On-Topic sessions

these session types provides valuable insights into user engagement patterns. Understanding the nature of these sessions helps in tailoring the recommendation system to cater to different user needs, be it for focused ideation or exploratory thinking. This analysis forms a foundation for enhancing the user experience on the Ideator platform by aligning the recommendation system with user preferences and how those preferences fit in the Supermind Design framework.

2.2.5 User Preferences and Personalization

A critical aspect of the analysis is the examination of user-provided preference data on Ideator's generation outputs:

Table 2.2: User Preferences Overview

Preference Type	Number of Users
Neutral Preferences	9852
Positive Preferences (Liked)	280
Negative Preferences (Disliked)	93

The minimal non-neutral preference data (3.6479%) poses a challenge for developing a personalized recommendation system.

2.2.6 Gaps and Opportunities

This analysis highlights several key gaps in the current Ideator platform:

- Low user retention past the first session
- Limited depth in user engagement per session
- Insufficient user-provided data for effective personalization

These findings will provide input in the design and development of the recommendation system aimed at enhancing user engagement, encouraging more in-depth ideation, and increasing the collection of personalization data, thereby improving the overall user experience on the Ideator platform.

2.3 Previous Recommendation Systems for Ideation

The paper, "Using Recommender Systems to Support Idea Generation Stage" by Maria El Haiba et al, gives important insight into the use of recommender systems for ideation. This paper explores the use of recommender systems in the early stages of innovation, specifically in generating ideas. The authors review various types of recommender systems, including Content-Based Filtering, Collaborative Filtering, Hybrid, Context-aware, and Social Network-Based systems. These systems are known for their ability to provide personalized suggestions, filter large information spaces, and handle user preferences efficiently. [6]

Recommender systems can nurture and drive idea generation by creating an innovative environment, encouraging employee participation, and leveraging knowledge. These systems are shown to support creativity, provide inspiration, and enable learning from a wide range of ideas, which is crucial for the ideation process. [6]

However, there are several challenges that these systems face, such as sparsity, cold start problems, scalability issues, over-specialization, and privacy concerns. Another noted challenge is the 'gray sheep' problem, where some users' preferences don't align well with any group, making recommendations difficult. [6]

For the recommendation system I am proposing for the Ideator Platform, these findings are particularly relevant. The following proposed system could integrate aspects of these various approaches, focusing on enhancing creativity, knowledge sharing, collaboration, and learning. It will also consider the context in which ideas are generated, ensuring recommendations are relevant and aligned with organizational goals. Addressing the challenges highlighted in the paper, such as sparsity and cold start problems, will be key to developing a robust and effective system. The goal is to create a system that not only suggests ideas but also stimulates further creative thinking, taking into account the unique requirements of the ideation process.

Chapter 3

Recommendation System Design Methodology

3.1 Research Objectives, Workflow, and Data

The following sections illustrate the initial implementation of a recommendation system developed using the Supermind Design methodology, informed by an analysis of user interactions on the Ideator platform. The primary goals are to establish a framework for the recommendation system, incorporate any overlapping features, solicit feedback through human studies, and refine the system based on the feedback received.

The data utilized to design the recommendation system and train its internal machine-learning models is sourced from user interactions on the Ideator platform. Prior to processing, the dataset consists of 24,008 data points, each characterized by the features described in Appendix A.

3.2 Version 1 of Recommendation System (static rules-based)

When engaging in open-ended creative problem-solving tasks, it is rare that the first ideas generated truly encapsulate the problem at hand or that the first solutions really accomplish the goal of the problem solver. For this reason, workshops often guide practitioners through a process with a facilitator who probes and pushes participants along through several steps to encourage reflection and iteration. This idea of encouraging more critical thinking from users is at the heart of the Supermind Design Methodology [7]. However, with the Supermind Ideator platform, a system that seeks to augment and scale up access to Supermind Design, no facilitator is present. When left to use the platform freely, we have observed that users often stay quite shallow in their use of the system. As such, we hypothesized that developing a 'recommender' that could encourage a user to keep going through the ideation process would be fruitful.

The first version of the recommender was developed using a static rules-based machine which takes in a user's problem and prior moves and then suggests what move they should use next to continue their ideation process.

3.2.1 Recommendation System Overview

The static rules-based recommendation system for the "next move" utilizes the principles laid out in the Supermind Design Primer by CCI, along with the structured innovation approach known as the double diamond design process. These frameworks serve as the foundational methodologies underpinning the system's development, promoting the use of collective intelligence within the innovative problem-solving and decision-making process.

The idea of a Supermind, as described in the Supermind Design primer, is a collectively intelligent system made up of individuals and/or machines that exhibit intelligent behavior through interactions with one another. This idea is what the recommendation system is attempting to help users harness throughout the ideation process by suggesting strategic Supermind design moves based on their current state in the problem-solving journey. This recommendation system takes place in two of the three main stages of Supermind Design: defining the problem and generating ideas, thereby facilitating a comprehensive exploration of the problem and solution space.

The Double Diamond design process complements the Supermind Design approach by emphasizing a structured path through discovery, definition, development, and delivery (see figure 3.1) [1]. This process advocates for a balanced approach to problem-solving that oscillates between divergent and convergent thinking—initially broadening the scope of exploration to consider a wide array of possibilities and subsequently narrowing down to specific, actionable solutions. The recommendation system supports this philosophy by incorporating specific moves—such as Zoom In, Zoom Out, Analogize, Groupify, Cognify, and Technify—which collectively enable users to traverse a wide spectrum of idea generation and refinement strategies. In addition to this, the recommendation system takes in to account the user's current problem statement and identifies its "scope" as an indicator of where a user is in their process of ideation in relation to converging or diverging (explained more in section 3.3).

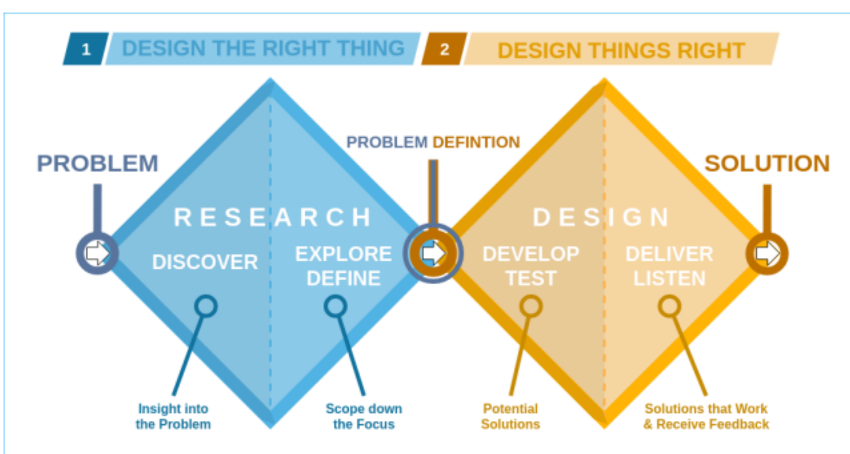

Figure 3.1: Illustration of Double Diamond process.

Each displayed move is recommended in a sequence that facilitates both the expansion and contraction of the problem and solution space as needed. For example, the Zoom Out move provides users with a more generalized perspective of the problem or solution space, which can help unveil new avenues of exploration. On the other hand, the Zoom In move pushes users to dive deeper into the specifics of their particular problem statement, encouraging a more focused analysis of their problem or solution. This dynamic between generalization and specification is prominent in the Supermind Design process, underscoring the importance of flexibility and adaptability in innovative problem-solving.

Moreover, the recommendation system's sequential logic for transitioning between different states based on the current problem scope and state is indicative of the iterative nature of the Supermind Design framework. It showcases an advanced understanding of how innovative solutions can be developed through repeated cycles of ideation and evaluation.

The following illustration is an overview of overarching rules used by the recommendation system to guide users through the ideation process:

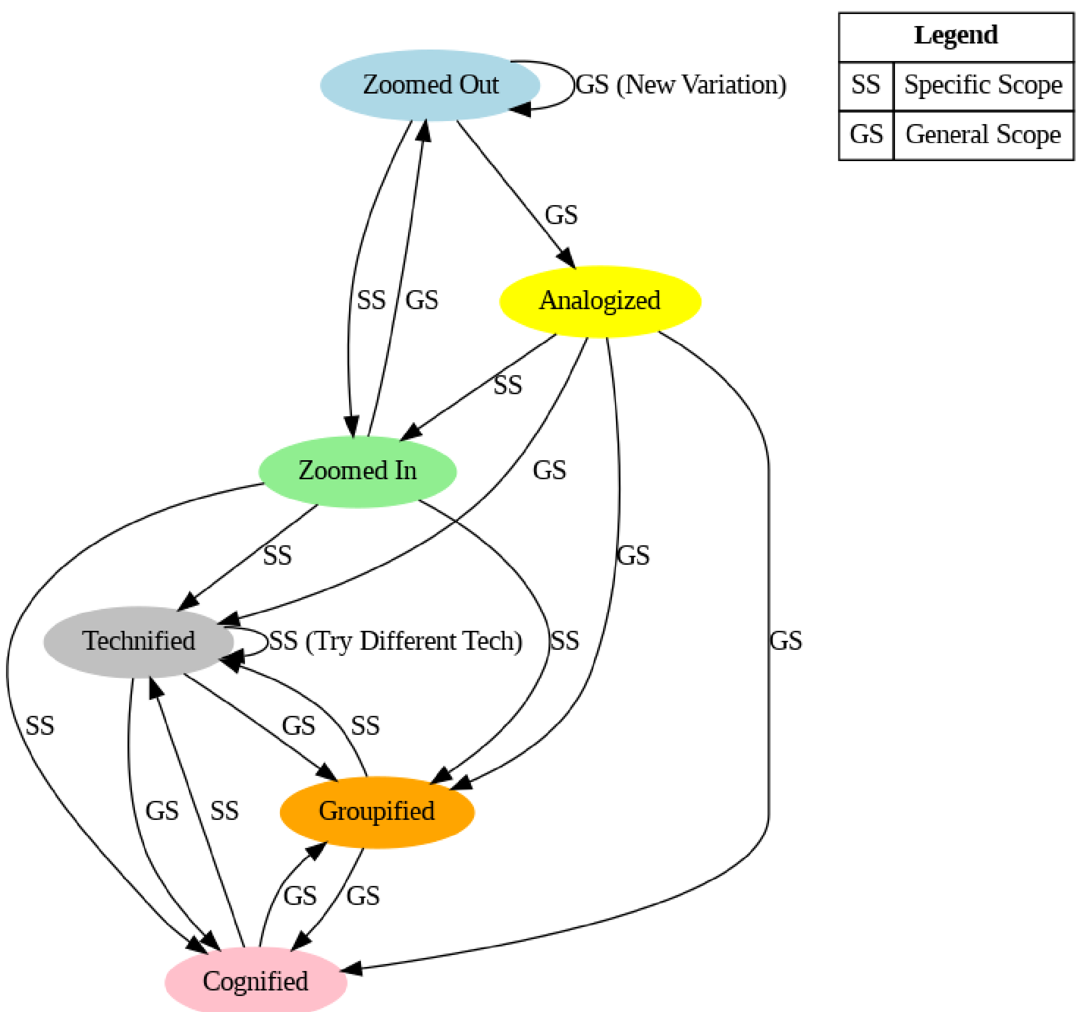

Figure 3.2: Overview of Recommendation System Rule Set

1. If previous move is Zoom Out (either in "Parts" or "Types"):
   For a "specific" problem scope, the recommendation to "Zoom In" aligns with the methodology's approach to deepen understanding by focusing on specific components or categories of the problem. This supports a more detailed and focused analysis which can uncover nuances that broad overviews might miss. For a "general" problem scope, suggesting moves like "Analogize" is consistent with the intent to expand creative thinking and explore diverse perspectives. The Supermind Design methodology advocates using analogies and reframing techniques to broaden the conceptual space and stimulate novel ideas, making these recommendations apt for users needing broader context.

2. If previous move is Zoom In (either in "Parts" or "Types"):
   When the problem scope is "general", recommending to "Zoom Out" helps reconnect the detailed insights back to the broader themes or overarching categories. This can facilitate integration of specific findings into larger contexts, which is a fundamental aspect of effective problem-solving as discussed in the Supermind Design methodology. For a "specific" problem scope, moving towards Supermind moves like "Groupify", "Cognify", and "Technify" follows the methodology's guidance for leveraging collective intelligence and technological tools to refine and implement solutions tailored to specific challenges.

3. If previous move is Analogize:
   For "specific" problem scopes, Zoom In is suggested as focusing on the details of the analogous ideas helps in extracting actionable insights and understanding deeper correlations, which is a critical step after broad ideation. For "general" problem scopes, proceeding with Supermind moves to plan and execute ("Technify", "Groupify", and "Cognify") based on the analogized concepts reflects the methodology's emphasis on using insights from analogies to inform strategic decision-making and solution design.

4. If previous move is Groupify:
   In "general" problem scopes, enhancing group decision-making capabilities through "Cognify" supports the methodology's focus on improving collective cognitive processes. In "specific" problem scopes, using "Technify" helps integrate relevant technologies into group tasks, aligning with the methodology's recommendation to leverage technology for enhancing efficiency and effectiveness.

5. If previous move is Cognify:
   If problem scope is "general", consider moving to "Groupify". This approach aligns with the Supermind Design methodology's emphasis on leveraging collective intelligence after enhancing individual cognitive processes. It ensures that the insights generated through enhanced cognition are utilized in a collaborative environment, maximizing their impact and facilitating comprehensive solutions. If problem scope is "specific", consider moving to Technify. This follows the methodology's guidance to integrate technology directly after cognitive enhancements, providing technical solutions that are specifically tailored to the detailed requirements identified through

cognitive processing. It ensures that the solutions are not only conceptually sound but also practically feasible and effectively implemented.

6. If previous move is Technify:
   If problem scope is "general", consider returning to "Groupify" or "Cognify". This cycle back to cognitive or group processes after implementing technological solutions ensures continuous refinement and enhancement of solutions through human and collective input. It aligns with the iterative nature of the Supermind Design methodology, which advocates for ongoing evaluation and adjustment of solutions. If problem scope is "specific", rerun "Technify" with other technologies. This step ensures that the technology in use is adequately meeting the specific requirements of the problem. Trying a range of existing technological tools helps determine if additional or alternative technologies are needed to better address the nuances of the specific issue.

In summary, the design and structure of the recommendation system are an evolution of the robustness of combining the Supermind Design approach with the double diamond design process. By embracing the principles of collective intelligence and structured innovation, the system provides a versatile tool for navigating complex problem-solving scenarios. It aims to show how theoretical frameworks can be applied to develop practical solutions on the Ideator platform that facilitate creative thinking and decision-making for many different topics.

3.3 Incorporating Problem "Scope"

The previously mentioned static rules-based system builds upon the idea of divergent and convergent thinking (or the Double Diamond method [8]), and building upon this foundation, the implementation of problem "scope" within the system—categorized as either "specific" or "general"—emerges as a critical mechanism for tailoring recommendations to the nature of the input problem statements. This distinction in problem scope is not just a classification but a strategic indicator that influences the trajectory of the recommendation path due to its close correlation to the previously mentioned ideas of convergence and divergence.

In terms of the ideation process when utilizing Ideator, "scope" can be described as one of the following:

- *General:* Containing the main features or elements of something without going into thorough detail/description.
- *Specific:* Precise and clearly defined statement(s) pertaining to a particular topic/problem.

The idea behind maintaining a succinct definition for "scope" is to only provide a loose framework for what "scope" is as to allow human labelers to contribute their own interpretation of whether a statement is "general" or "specific". This looser framework also works to tie "scope" to the Double Diamond process where a "general" scope can correspond (not necessarily directly) to diverging ideas while a "specific" scope can be related to converging ideas.

Through analysis of user behavior with the Supermind Ideator, we can begin to approximate an understanding of their current state and trajectory through the problem-solving process. While we can not know with certainty the user's cognitive state or current conceptual framing, behavioral cues like the specificity of the problem they input to the system can hint at how developed their current understanding is. We aim to use this to align what sort of next steps we recommend. The following section will delve into the nuances of how this idea of problem "scope" is implemented and integrated within the recommendation system to provide more informed recommendations.

3.3.1 Intuition Behind Unsupervised ML Approach

Given a present lack of prior work pertaining to 'scope' analysis or 'specificity' classification, we develop a novel pipeline that is informed by the same heuristics humans use when determining whether or not some string of text is 'specific.' This approach combines several ML and NLP tactics to implement a classifier that can evaluate any string.

3.3.2 Implementing Approach/Experimentation

The overarching approach for determining the "scope" of a user's input problem statement is the following:

1. Embed the textual data so that it can be analyzed effectively.
2. Create clusters (and sub-clusters) based on "topic" to analyze problems focused on similar topics.
3. Utilize general language-based heuristics to create a separate embedding of problem queries within their "topic" cluster based on "specificity." (Discussed further in section 3.3.2.2)
4. Cluster problem queries based on their heuristic embedding given the "topic" cluster they fall into.

In order to determine the specificity/scope of a user's input, we must first understand how each input relates to the other 2427 unique problem inputs created by other users. By embedding all of this text information and creating initial clusters, we can reduce the evaluation space to only those topically/thematically relevant items. Given this reduced comparison space, we can then evaluate content across a number of linguistic and syntactic heuristics to construct another embedding of the problem that pertains to topic and specificity. Finally, these heuristics embeddings can be used to help classify the specificity of novel input based on topic relevance.

Topic Clustering and Sub Clustering

To first understand what topics and thematic spaces our data falls into, we developed an approach that uses KMeans to cluster strings. The initial step involved conducting a main clustering pass where the general topics were identified. Notably, there was a consistent trend of one larger cluster emerging, which contained a majority of the queries. Given this occurrence, a secondary KMeans clustering was performed specifically on this large cluster to achieve refined topic sub-clustering. This two-layered approach allowed for both broad and more nuanced insights into the dataset's thematic structure, which is explained further in section 3.2.2.1.2.

Justification for Using KMeans:
To cluster the problem input data correctly, KMeans was selected for its efficacy in handling vector space representations of text data, a critical feature given that our data were embedded using Doc2Vec. The choice of KMeans over other clustering methods like DBSCAN [9] or hierarchical clustering [10] was driven by several factors:

Vector Space Similarity: KMeans excels in environments where data points (in this case, text) are represented in vector spaces. Doc2Vec embeds text in a high-dimensional space where semantic similarities translate to spatial proximities (explained in more detail in section 3.2.2.1.1). KMeans leverages this attribute by grouping texts to minimize intra-cluster variance, effectively capturing the essence of topics within clusters. Simplicity of Interpretation: The centroids in KMeans, which represent the mean vector of all points in a cluster, provide an intuitive understanding of each cluster's thematic focus. This straightforward interpretation is particularly beneficial in text clustering, where identifying the central theme quickly is advantageous.

The effectiveness of KMeans in this context is attributed to its reliance on the Euclidean distance to measure the similarity between data points, which works well in vector spaces. By calculating the centroid of assigned points for each cluster and iteratively optimizing these centroids, KMeans ensures that each cluster is as compact and separate as possible, which is ideal for discerning distinct topics in a collection of text documents [11].

Feature Representation (Doc2Vec)

In my exploration of feature embeddings for text data, I opted to employ the Doc2Vec model, also known as Paragraph Vector, due to its robust capabilities in handling larger blocks of text. Developed by Le and Mikolov in their 2014 paper "Distributed Representations of Sentences and Documents," [12] Doc2Vec extends the Word2Vec model to generate embeddings for phrases, sentences, paragraphs, or entire documents rather than just individual words. This advancement is particularly important for this study as it can capture the semantic meaning of entire documents (input problem statements that can be multiple sentences long). Doc2Vec utilizes two primary architectures: Distributed Memory (DM) and Distributed Bag of Words (DBOW) [12]. Like Word2Vec's Continuous Bag of Words model, the DM model predicts a word based on the words around it in a given context and uniquely integrates a document-specific vector. The mathematical objective of DM can be described as maximizing the log probability

$$max(\log_p(w_t|w_{t-k},...,w_{t+k},d))$$

where w_t represents the target word, $w_{t-k},...,w_{t+k}$ denote the words within the context window k, and d is the unique document vector. Conversely, the DBOW model parallels Word2Vec's Skip-gram model by ignoring the context words and predicting words randomly sampled from the paragraph using only the document vector, optimizing the sum

$$max(\sum_{w \in context} (\log_p(w|d))$$

The choice to utilize Doc2Vec is justified by several factors crucial for the semantic analysis of longer texts. First, its contextual awareness, which includes a unique document identifier, allows it to comprehend and encode the broader context rather than merely focus on local word relationships. This feature makes it particularly adept at capturing the thematic essence of lengthy documents (or, in this case, input problem statements). Additionally, Doc2Vec provides a fixed-length output from variable-length texts, a necessity for standardized input sizes in modeling tasks. This model's ability to maintain semantic similarity across a document is invaluable for applications such as document similarity and retrieval, where understanding the holistic thematic structure is paramount.
Given these characteristics, Doc2Vec stands out as a superior choice for text embedding over simpler models like TF-IDF [13] or even Word2Vec [14] when dealing with extensive textual data. This ability to encapsulate broader narrative flows into a singular vector representation of a document's meaning significantly enhances the capability of analysis of input problem statements, allowing for better clustering and semantic analysis of large-scale document (problem statement) corpora.

Experimentation (Random Seeds Different Clustering Approaches)

Throughout the course of my experimentation, I utilized KMeans clustering to identify inherent groupings within a dataset of embedded textual queries. It is important to note that KMeans is sensitive to the initial random seed used to start the algorithm, which influences the selection of initial centroids and can lead to different clustering results. To ensure the reproducibility of my results, I set a specific random seed before each clustering operation. This practice is critical for scientific rigor, allowing others to replicate the experiment and achieve the same cluster assignments for given inputs.

I initially applied KMeans clustering directly to the entirety of the query data. This method consistently resulted in forming one significantly larger cluster than the rest, as can be visualized in the graph titled "UMAP Visualization of Main Clusters with Summaries." This emergence of one large cluster suggested the prevalence of a dominant topic in the dataset, which did not provide a significant clustering of our data.

To effectively illustrate the clustering output, I employed Uniform Manifold Approximation and Projection (UMAP). UMAP is a novel manifold learning technique for dimensionality reduction. UMAP operates on the principle of topological structure preservation within high-dimensional data. It begins by constructing a weighted k-nearest neighbor graph representing each data point's local neighborhood. Then, UMAP employs fuzzy set theory–a mathematical method for decision-making that uses fuzzy descriptions of information–to convert distances in the high-dimensional space into probabilities, capturing the likelihood that two points are connected. By optimizing the layout of these points in a low-dimensional space, typically two or three dimensions, UMAP seeks to approximate the high-dimensional topological structure.

The optimization process in UMAP is based on minimizing the cross-entropy between two fuzzy topological representations: the high-dimensional original data and the low-dimensional projection. The mathematical foundation of UMAP allows it to preserve both local and global data structures, making it particularly well-suited for visualizing clusters and the relationships between them. This aspect is beneficial for identifying patterns and structures that are not immediately apparent in high-dimensional spaces.

I selected UMAP over other dimensionality reduction techniques, such as t-SNE [15] or PCA [16], for several reasons. Unlike PCA, which is a linear projection method that may not capture the nonlinear relationships inherent in text data, UMAP can effectively model the nonlinear manifold on which the data may lie. Compared to t-SNE, UMAP tends to preserve a better global structure, which is essential when interpreting the overall arrangement of clusters. Additionally, UMAP's ability to scale to larger datasets and its less stringent hyperparameter tuning process made it a more practical choice for the task. The consistency of UMAP's dimensionality reduction facilitates a more nuanced understanding of the clusters and their interrelationships, thus providing an invaluable tool for visualizing the complex structures within the text data derived from my analysis.

The primary clustering provided useful insight (see figure 3.3): the largest cluster could be further analyzed. I proceeded with sub-clustering this main cluster to obtain more granular thematic divisions within it. Figure 3.4 demonstrates the result of this sub-clustering. By reapplying KMeans to this predominant cluster, the data revealed a finer granularity of subtopics, allowing for a more detailed exploration of the thematic landscape. As seen in the visualization, these refined sub-clusters exhibit a more balanced distribution and provide a deeper understanding of the thematic variety within the larger context, underscoring the efficacy of sub-clustering in extracting nuanced thematic information from dominant clusters. This result proved to be useful down the line when determining problem input "scope."

Heuristic-Based Clustering

In this section, we introduce a heuristic-based approach to clustering within the domain of natural language processing (NLP), specifically designed to discern the "scope" of textual content—an area not conventionally addressed by existing NLP techniques. "Scope" on the Ideator platform is understood as the focus of an input problem statement. This concept is central to the effectiveness of our recommendation system, which aims to align the suggested next move with the user's path of convergence or divergence on a particular problem or solution.

The heuristic-based clustering methodology leverages features such as readability scores, meaningful and unique word counts, numerical data frequency, average sentence length, and lexical diversity. We aim to categorize content by its inherent scope by clustering texts based on these heuristic indicators. This novel approach enhances the recommendation system on the Ideator platform, ensuring that users are pushed in a helpful direction given the Supermind Ideation methodology.

Given the previous section which utilizes Doc2Vec to cluster input problem texts by topic, the following heuristic features are utilized to cluster the queries to classify "scope". The idea behind clustering the problem queries by topic is to separate all the information into groupings where the content is similar between each query. The topic separation is

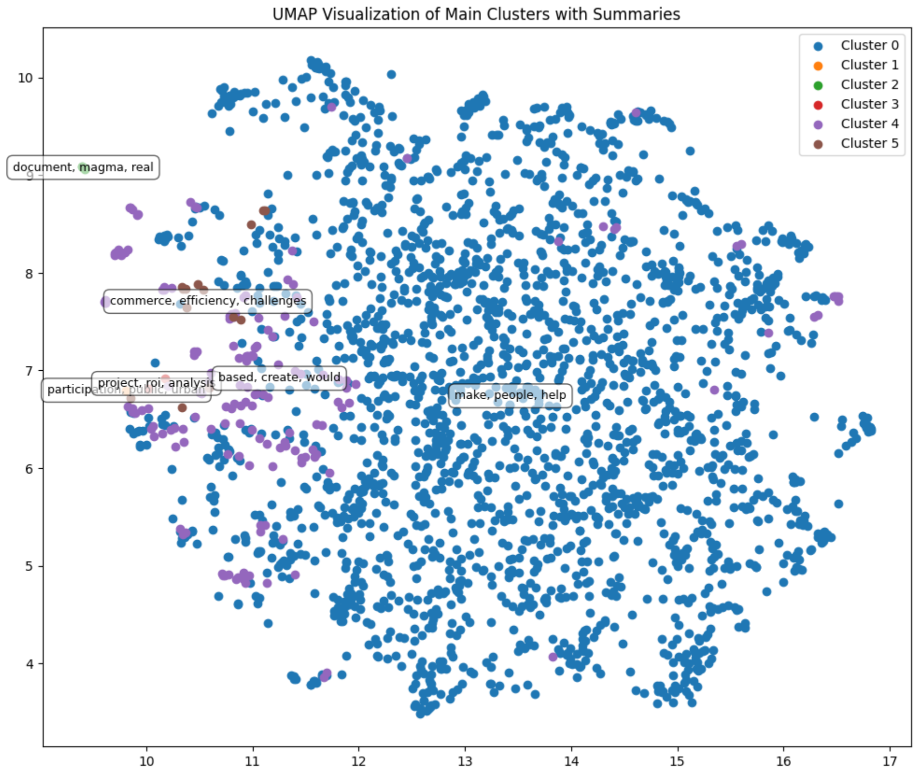

Figure 3.3: Visualization of Main Topic Clusters

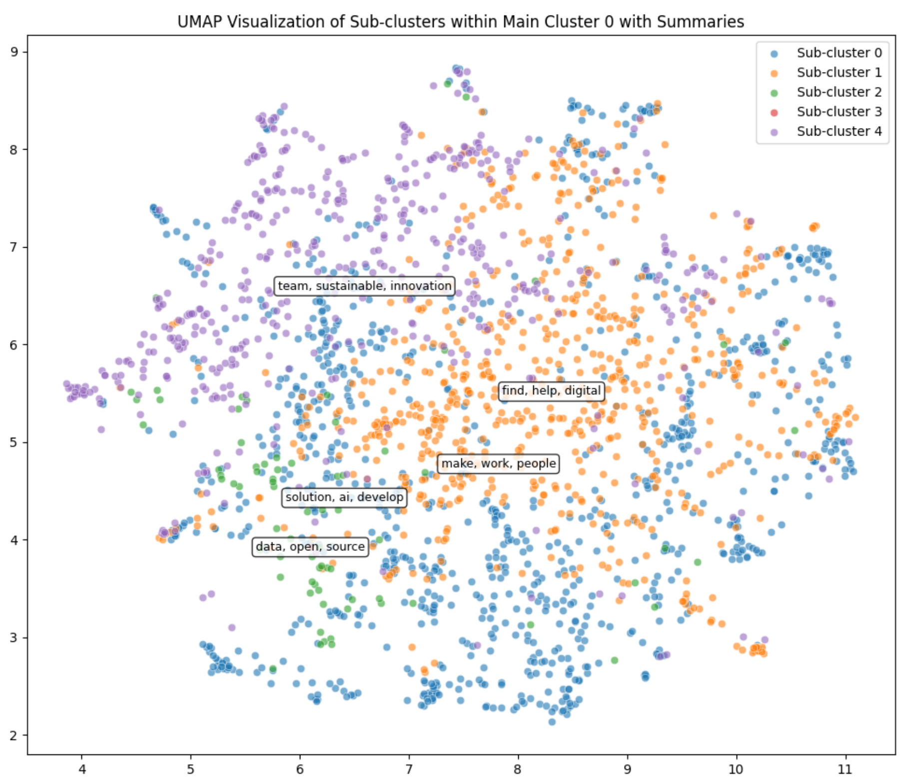

Figure 3.4: Visualization of Sub-Topic (Refined) Clusters for Largest Main Cluster

important (supported by results in section 3.4) as merely using the heuristic features for clustering on the raw problem queries does not provide us with a useful classification. Along these lines, the more refined the clusterings are (processing sub-clusters after creating main topic clusterings), the more accurate the final classification results will be. Creating query clusters with highly related topics allows the heuristic features to further break down the queries in terms of their syntactical structure to determine “scope”.

Feature Representation

Building on the heuristic-based clustering framework outlined in Section 3.2.2.2, the choice and representation of features are pivotal for inferring the scope of textual content. Each selected feature serves as a heuristic cue to align the users’ ideation path within the Ideator platform with the current input problem’s “scope.”

Incorporating Named Entity Recognition (NER), utilizing the spaCy ‘en-core-web-lg’ model, is foundational in extracting named topics, issues, and other critical entities in texts, such as articles. This method not only locates but categorizes important nouns and proper nouns—key indicators of the subject matter discussed. The resultant entities, linked by their co-occurrence and visualized as networks, serve as nodes in the broader thematic structure of the text. The spaCy model is favored for its efficiency and simplicity over more complex models, such as those offered by transformer pipelines [17].

The Flesch-Kincaid Reading Ease formula is applied to compute readability scores, a determinant of textual accessibility. This metric assesses how effortlessly a text can be understood through the following formula.

Readability Score = 206.835 - (1.015 $\times$ Average Sentence Length) - (84.6 $\times$ Average Syllables Per Word).

The interplay between syllable count and sentence length yields a score that helps classify texts into broader or more specialized categories. For instance, a high readability score often correlates with a wider scope, suitable for general audiences, while a lower score may imply specialized content for a narrower audience segment [18].

Furthermore, the count of meaningful words, those not labeled as common stop words, indicates the richness of content and its topical specificity. The unique word count acts as a proxy for lexical diversity, where a greater count suggests a text covering a wide array of topics [19], [20]. Conversely, a lower count might signify a more concentrated thematic focus.

The prevalence of numerical data within a text is another heuristic, hinting at a text’s precision and technical nature, potentially signifying a concentrated scope [21]. Similarly, the average sentence length, with longer sentences typically bearing more complex structures, may cater to an audience with a higher level of expertise, indicating a narrower scope [22].

Lastly, lexical diversity is measured to gauge the range of vocabulary employed—a higher diversity can point to scholarly or creative texts targeted at a more selective readership [19], [23]. In contrast, texts with lesser lexical diversity are possibly directed toward a broader audience, covering more generalized topics.

The picture painted by these heuristic features allows for an insightful clustering of texts based on “scope.” This ties back to the Supermind Ideation methodology, ensuring that the Ideator platform’s recommendation system is not only precise but also contextually sensitive to the user’s convergence or divergence path.

To reiterate, our unsupervised machine learning approach is doing the following:

1. Embedding the problem input statements of every user utilizing the Doc2Vec embedding model.

2. Clustering problem input statements by topic and the sub-topic through the use of the KMeans algorithm.

3. Our system then parses each input problem statement by each independent topic cluster (as to only compare problem statements from the same topic cluster), transforms the problem statements into a feature vector represented by our heuristic feature representation, then utilizes KMeans to form clusters with the new heuristic feature representation.

4. After utilizing KMeans to create heuristic based clusters for each topic cluster, we deem problem statements from the majority cluster (cluster with most example in the heuristic based clusters) as “general” in scope and the minority as “specific”.

3.4 Analysis of Human Feedback/Labeling

Structure of Experiment

After implementing the unsupervised learning approach to determine the “scope” of input problem statements, our lab performed an experiment to gain human feedback on the approach. The study aimed to provide candidates with problem statements created by users on the Ideator Platform so that they could classify statements as either being “specific” or “general” in scope. We intentionally did not provide much guidance on how the users should deem the specificity of scope to avoid any influence on their evaluation process. The statement below was shown before any problem statements were displayed for them to label:

“This survey will present you with several lists of problem statements that have been written by people before they try to come up with creative solutions to their problems.

Your task is to rate each of these problem statements as either being **General** (meaning "containing the main features or elements of something") or **Specific** (meaning "precise and clearly defined").

If you are unsure, you can select **Don’t Know**.”

After the previous message was shown, the users were taken to the next page, where problem statements were displayed for them to label, as shown in the figure 3.5.

The problem statements shown on each page fall under the same topic clustering, so users do not have to jump between topics when determining “scope,” similar to how the unsupervised ML approach receives data in terms of topic clusters before creating heuristic-based clusters of the data to determine “scope.” We also keep track of the labels (“General” or

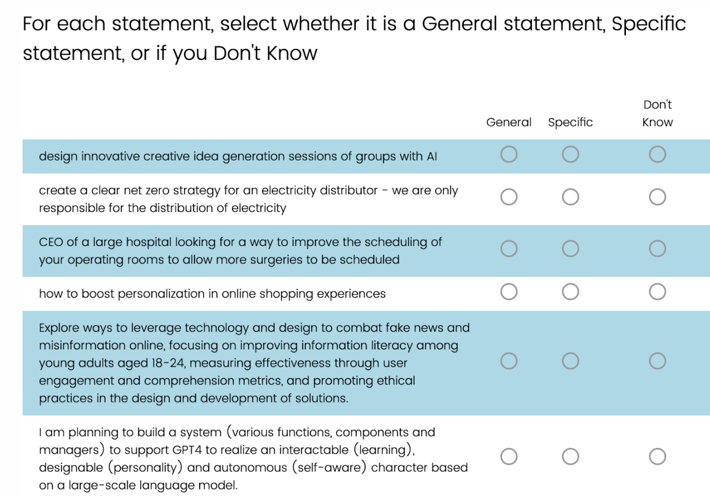

Figure 3.5: Demonstration of how users were tasked with labeling problem statements.

“Specific”) created by the unsupervised ML approach to compare the results with the human labels created.

Human Feedback Metrics

The following section analyzes the initial effectiveness of our unsupervised learning approach in classifying problem statements into ’General’ or ’Specific’ categories as we analyze the human feedback obtained through our experimental survey. The results from the survey are illustrated in three distinct parts.

Firstly, as depicted in the figure 3.6, the distribution between ’General’ and ’Specific’ labels created by humans is fairly even, with no significant difference between them (ns), indicating a balanced perception among users in distinguishing these categories. However, both categories differ significantly from the ’Don’t Know’ option, suggesting that participants were confident in their ability to classify the statements according to the provided definitions.

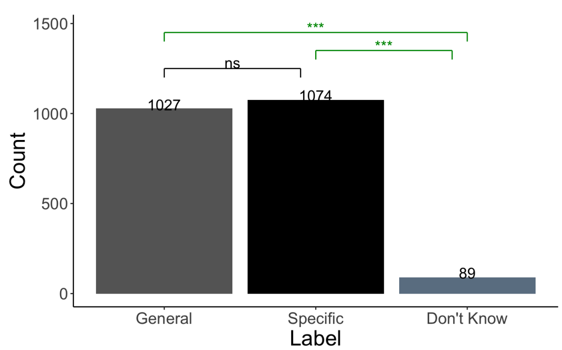

Figure 3.6: Distribution of ’General’ and ’Specific’ human labels.

Secondly, the accuracy of the classifications was assessed against the heuristic labels generated by the unsupervised model (see figure 3.7). The statistical analysis demonstrated a significant tendency for classifications to align with the model's labels, indicating overall high accuracy. Specifically, a logistic regression analysis revealed that classifications of 'General' labels were more likely to be correct compared to 'Specific' labels. This suggests that the heuristic approach is particularly effective at correctly identifying statements with a broader scope, as evidenced by the higher accuracy in 'General' classifications. Also, humans were generally significantly more likely to align with the model's label than they were to misalign. The graph below illustrates the classification results from our study, showing 1,296 correct and 894 incorrect classifications. It indicates a statistically significant higher accuracy for 'General' labels over 'Specific' ones, with a p-value less than 0.001.

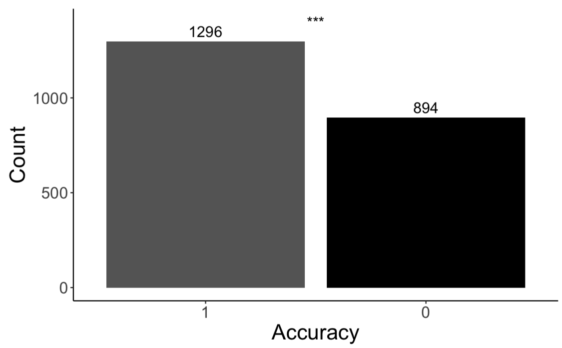

Figure 3.7: Accuracy of human labels when compared to system produced labels.

Thirdly, further validation of the heuristic labeling accuracy was provided through a detailed comparison of the labels where the consensus was clear (i.e., there was a majority of human labels for one category). The analysis evaluated the accuracy of a heuristic model by comparing its labels with a consensus label derived from the majority of human judgments on 88 problem statements. The graph below shows that in cases with a clear majority consensus among humans, the heuristic model's labels were correct 62 times and incorrect 26 times. This resulted in a statistically significant chi-squared test outcome ($p = .0001$), indicating that the heuristic model aligns well with the majority human judgment. The significant p-value confirms the model's reliability in accurately reflecting human consensus on these labels.

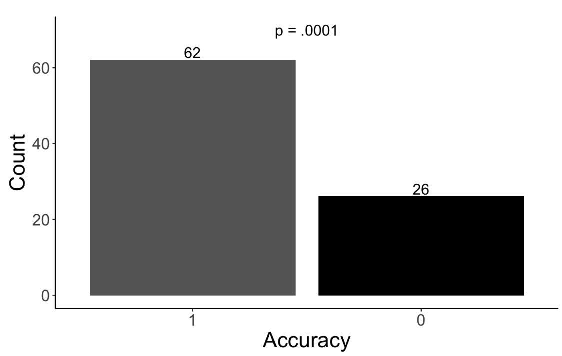

Figure 3.8: Direct human label accuracy when there was a distinct consensus label.

Finally, there is an intraclass correlation coefficient (ICC) of .83, reflecting high consistency among raters, which underpins the reliability of the human judgments made during the experiment. This strong ICC indicates that despite the subjective nature of the task, there was a high degree of agreement among participants, reinforcing the experiment's validity and the usability of the unsupervised model in practical scenarios.

Together, these metrics not only validate the model's performance but also illustrate its potential applicability in environments where discerning the scope of problem statements is crucial for downstream processing or decision-making.

Reoptimizing Unsupervised Model

Building on the insights gained from the initial analyses of our unsupervised learning model's performance, we turned back our focus to refining and optimizing the model's accuracy further. Despite the model's proven ability to align well with human judgment, especially in categorizing 'General' statements, there was still room for improvement in its precision and consistency. Recognizing the sensitivity of the KMeans clustering algorithm to initial starting conditions, we explored the impact of varying random seeds. The subsequent section details our approach to reoptimizing the model by testing different random seeds to identify the best accuracy with the consensus human labels. This method aims to enhance the accuracy and robustness of the classifications, ultimately striving for even higher alignment with human judgment. Note that the following statistics illustrate varying approaches to creating heuristic ("scope") labels on the problem statement data, where "Match Percentage" is the number of heuristic labels generated by the system that match human consensus labels.

Table 3.1: Optimization Results Comparison

Method	Mean Match Percentage (%)	Std of Match Percentage (%)	Best Match Percentage (%)
Heuristic Clustering Only	44.32	0.00	44.32
Main Clustering + Heuristic	69.14	2.19	71.59
Complete Main and Sub Clustering + Heuristic	72.58	3.10	80.64

The outcomes suggest that the final approach, which combines the main and sub-clustering (based on topics) with the heuristic feature clustering, performs the best. Essentially, when comparing problem queries arbitrarily without understanding topics, there is less to draw from (in terms of heuristic features) as the essence of each problem query is more random, whereas topic clustering (and refining topics by sub clustering) ensures that each problem query is at least related in terms of the topic of focus.

Also note that there is some standard deviation of accuracy in the results as the process of finding the best starting state involves choosing a random seed, which inherently introduces variability due to KMeans non-determinism [24].

Chapter 4

Experiment, Evaluation Discussion

After developing the Ideation “next move” recommendation system, an important step is to assess the effectiveness of these theoretical enhancements. The following sections delve into the structure and analysis of a structured pilot experiment designed to gain insight into how raters view sequences generated by the recommendation system.

4.1 Experiment Structure

The experiment is designed as a blind study to gain insight from raters on how they view the recommendation system. Participants are shown 10 different sequences of ideas; each presented on a separate webpage to reduce direct comparison, that are generated either by randomly selecting moves or utilizing moves selected by the recommendation system. The origin of each sequence—whether random or system-generated—is not disclosed to the raters, ensuring an unbiased evaluation environment. All sequences commence with the "Zoom In Parts" move to standardize the starting point and are capped at 5 ideas to maintain consistency and manageability across all tests. This structured approach allows for a controlled comparison of the system’s output against random generation.

It is important to note that the initial input problem statement is “How can I discern fake news from real news?” and the subsequent information that given the prior messages from the thread of ideas, we utilize an LLM to "Pick a single point from prior output" so the input problem can be changed to anything from the thread of previous outputs (in this case, previous ideas). This inherently leads to variability in the subsequent outputs, meaning no two sequences generated by random moves or by the recommender will be deterministic.

An example of an idea sequence that would be shown to a participant is reflected infigure 4.1 on the following page.

4.2 Evaluation

4.2.1 Assessment Criteria

Two qualitative metrics are provided to users to guide their rating of the sequences they were given:

Review the ideas below ***as a sequence of steps*** for solving the following problem:

How can I discern fake news from real news?

#	Idea
1	- Verify the news source - Cross-reference information with other reliable sources - Check for bias in reporting - Evaluate the credibility of the author/reporter - Look for supporting evidence or sources - Be skeptical of sensationalist headlines or stories
2	ensure accurate information
3	1. **Analogies for detecting bias in reporting**: a. Checking for bias in reporting is like examining the ingredients list on a food label - you need to look beyond what's displayed prominently to understand what's truly going on. b. Analyzing bias in reporting is akin to fact-checking a story with multiple sources - you need to cross-reference to get the full picture. c. Identifying bias in reporting is similar to solving a puzzle - you need to piece together different perspectives to see the complete picture. d. Detecting bias in reporting is like exploring a maze - you need to navigate through different paths to find the correct way out. 2. **Reminder**: Always consider the source, language used, omission of key facts, sensationalism, and potential conflicts of interest when evaluating bias in reporting.
4	1. Engage community members in media literacy workshops to identify bias in reporting 2. Establish community-led fact-checking groups to review and analyze news sources for accuracy and bias 3. Encourage open discussions within communities about the importance of diverse perspectives in news reporting 4. Support local journalism initiatives that prioritize transparency and unbiased reporting
5	To establish community-led fact-checking groups effectively, consider historical precedents like the rise of citizen journalism movements, the impact of digital misinformation on public discourse, and successful local initiatives promoting media literacy. Remember the importance of diverse perspectives, transparency in funding sources, and clear guidelines for fact-checking processes. Additionally, understanding the evolution of media regulations and the challenges faced by traditional news outlets can provide valuable insights into shaping sustainable and impactful fact-checking efforts within communities.

Figure 4.1: Example of Idea Sequence.

1. Innovativeness: How creative and useful was a particular output or entire sequence.
2. Fit: How well does a particular output or entire sequence align with addressing the problem statement.

After each idea sequence, participants are asked to evaluate each sequence based on how well the ideas "fit" the specific challenge of discerning fake news from real news, using a 1-7 scale where 1 indicates "Not Well At All" and 7 denotes "Extremely Well." Additionally, raters are asked to judge the innovativeness of the sequences, and this scale also ranges from 1 (Not At All Innovative) to 7 (Extremely Innovative). Furthermore, raters are prompted to select which single idea within each sequence they find best fits or is the most innovative in relation to the problem. These qualitative assessments help with providing insights into the recommendation system's capacity to foster effective and innovative thinking within structured ideation processes.

4.2.2 Human Evaluation Results

The following section delves into the findings from the pilot experiment, where participants rated the ideation sequences based on innovativeness and fit to the problem of discerning fake news from real news. By examining these ratings and the preferences for specific ideas within each sequence, we can glean deeper insights into how raters view the outputs produced by the recommendation system and how it compares to randomly generated sequences. This analysis is pivotal in understanding the practical impact of the recommendation system on enhancing ideation and problem-solving capabilities.

From analyzing the ratings given to sequences as a whole, one can see that the signal produced from the study is quite small as the average rating of sequences, for either random generation or from the recommendation system, is centered around the neutral rating "4" (see table 4.1 below).

Table 4.1: Human Ratings Overview

Condition	Item	Mean	Std
Random	Seq.Fit	4.853760	1.551373
Random	Seq.Inn	4.579972	1.388949
Recommender	Seq.Fit	4.608808	1.575534
Recommender	Seq.Inn	4.459250	1.431495

Although the previous results show there is a minimal signal, there is a slight trend that the participants rate the randomly generated sequences in terms of "fit" and "innovation" higher than the recommender sequences. Further analyzing the data by low and high ratings (ratings between 1-3 and 5-7, respectively), we can see that the distribution of ratings is almost identical for both the sequence generated at random and those generated by the recommendation system (see figure 4.2 below).

Another trend that arises when analyzing the rating for individual ideas (where the number corresponds to the position in the sequence), based on "innovativeness" and "fit," is that

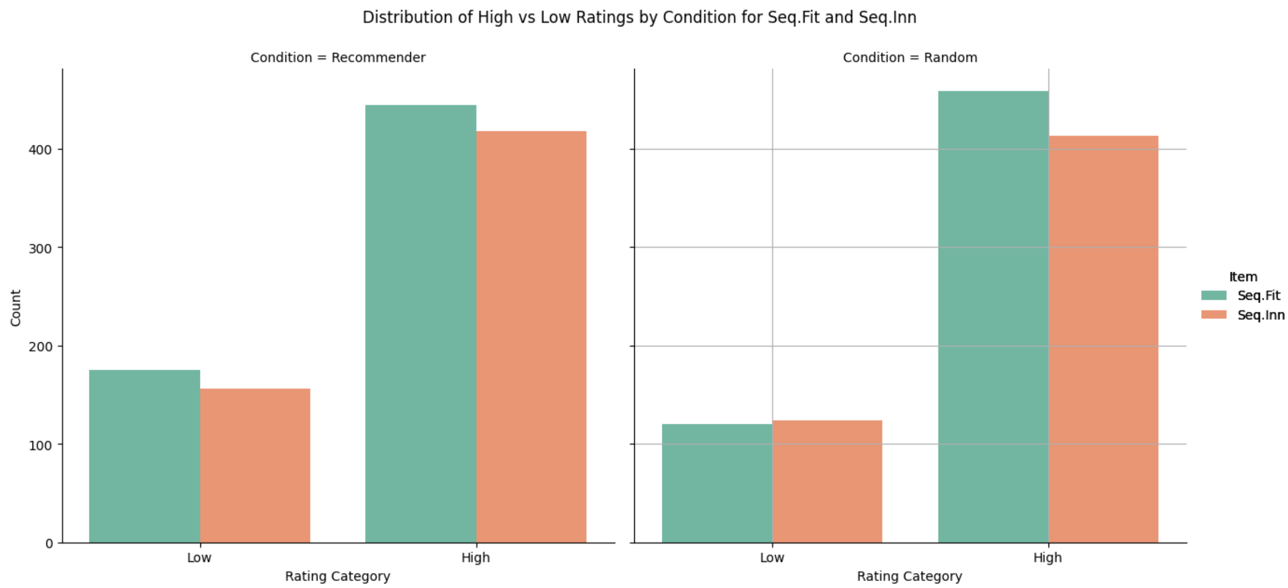

Figure 4.2: Distribution of ratings on entire sequence based on criteria.

the first idea was chosen the most times for having the best "fit." On the other hand, looking past the first idea, the number of ideas chosen based on "fit" seems to increase as the sequence progresses in the recommendation-generated examples, but the inverse for the randomly generated sequences. There also appears to be an almost perfectly normal distribution of idea selections in terms of "innovation" for the randomly generated sequences, whereas the distribution of selections is more skewed towards later ideas in the recommendation-generated sequences (illustrated in figure 4.3 below).

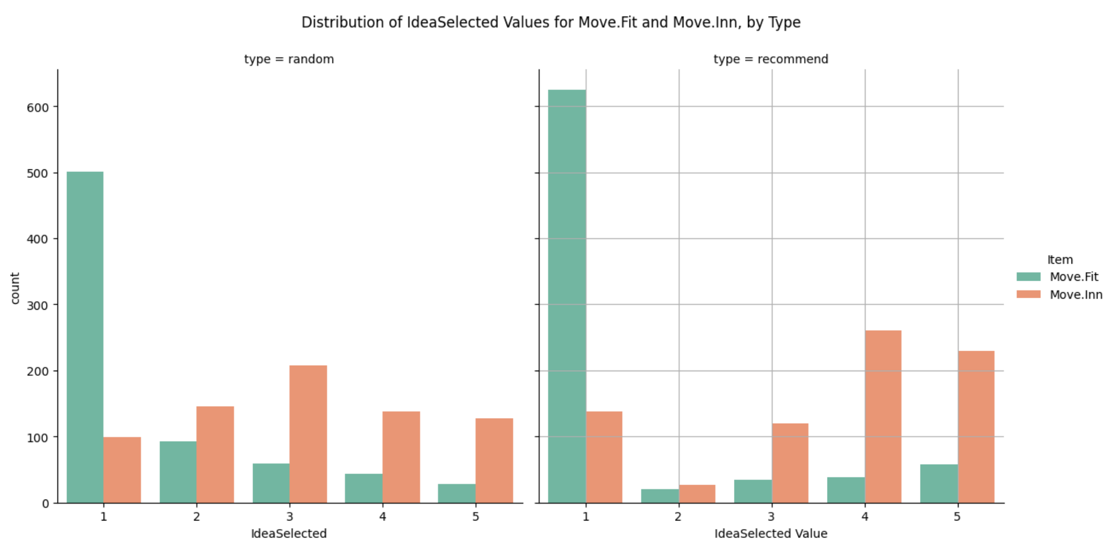

Figure 4.3: Distribution of idea selection in sequence based on criteria.

4.3 Discussion

4.3.1 Recommendation System Strengths

Despite the findings regarding the overall neutrality in sequence ratings, the pilot study exposes certain strengths in the recommendation system that are not immediately apparent through average ratings alone. Some of the strengths are highlighted below:

Progressive Relevance: The analysis suggests that while the first idea in sequences is strongly recognized for its fit to the problem statement, ideas generated by the recommendation system demonstrate a growing relevance and alignment as the sequence progresses (after the first idea). This suggests that the system has strength in progressively building upon initial ideas and refining the thought process, which could be crucial for users seeking to develop a deep understanding of complex issues like discerning fake from real news.

Enhanced Depth and Innovation in Later Stages: Furthermore, the skew towards later ideas in terms of innovativeness in sequences generated by the recommendation system indicates its capacity to introduce more creative and nuanced solutions as the ideation process unfolds. This finding is particularly valuable as it suggests that the system has the potential to increase its innovative output, providing users with richer and more varied perspectives the further they engage with the ideation sequence.

Structural Guidance: The underlying trends imply that the recommendation system provides structural guidance that may not initially outshine random generation regarding immediate impact (reflected in the neutral average ratings). However, it offers a consistent development of contextually relevant and increasingly innovative ideas, which is a subtle yet powerful tool for aiding users in complex ideation sessions. Moreover, the randomness of how random-generated ideas are chosen as best “fit” and “innovativeness” is a telling sign that there is no solid structure in how content is generated when not using the recommendation system. The methodical buildup when using the recommendation system could be instrumental in helping users not only understand their problem more thoroughly but also explore a broader range of potential solutions.

4.3.2 Recommendation System Weaknesses

The insights gleaned from the human evaluation results shed light on several weaknesses of the recommendation system, particularly when juxtaposed against randomly generated sequences. While the system aims to guide users through structured ideation effectively, the data suggests areas where it falls short or does not markedly outperform random generation.

Minimal Distinguishing Impact: One of the main weaknesses observed is the minimal distinguishing impact of the recommendation system compared to random sequences. The average ratings for both "fit" and "innovation" hovered around the neutral point, with the recommendation system occasionally scoring lower than the random sequences. This suggests that the system may not effectively leverage its theoretical underpinnings to produce significantly more relevant or useful outputs than random chance. There are also some structural intricacies in how sequences are currently being generated that could have been the cause of this issue, which will be addressed in section 4.3.3.

Innovativeness and Creativity Constraints: While the system is designed to foster innovative thinking, the ratings for innovativeness did not significantly favor the recommendation system over random sequences. This outcome could indicate a potential constraint in how the system stimulates creativity. It may be overly reliant on predefined paths or lack the ability to engage users with novel or surprising ideas, which are essential for sustaining innovation in dynamic problem-solving environments.

Addressing these weaknesses is essential to fully harnessing the potential of the recommendation system to enhance ideation and creative problem-solving on a broader scale.

4.3.3 Potential Further Experiments

Several additional experimental designs could be implemented to continue understanding the effectiveness of the ideation "next move" recommendation system and refine its utility compared to random generation. These experiments aim to deepen insights into user perceptions and interactions with the generated sequences, testing various sequence generation and presentation aspects.

1. Comparative Side-by-Side Evaluations A direct comparison experiment could be conducted where participants are shown two sequences side by side—one generated by the recommendation system and the other generated randomly—and asked to choose which sequence better addresses the problem statement. This approach can provide clearer insights into user preferences and the perceived effectiveness of the recommendation system over random selections.

2. Varying the Starting Move Rather than always starting sequences with the "Zoom In Parts" move, varying the initial move could offer insights into how different starting points influence the development and reception of idea sequences. This variation can help determine if the effectiveness of the recommendation system is contingent on the starting move or if it consistently aids users regardless of the initial ideation step.

3. Expanding Rating Metrics Additional metrics could be introduced to capture more dimensions of user experience, such as clarity, relevance, and actionability. Clarity could assess how understandable the ideas are, relevance could evaluate how closely ideas pertain to the main problem, and actionability could gauge the practicality and feasibility of implementing the suggested ideas.

4. Longitudinal Study Implementing a longitudinal study where users directly engage with the system over multiple sessions could provide insights into learning effects, system adaptability over time, and user satisfaction after prolonged use. This study would help in understanding how repeated interactions with the system influence user perception and creativity.

5. Alternative Problem Statements Testing the system with various problem statements across different domains could evaluate the system's versatility and effectiveness across a broader range of scenarios. This would also test the system's utility in generating relevant and innovative ideas in diverse contexts.

Each of these experiments could provide deeper insights into the strengths and limitations of the recommendation system, guiding further development to enhance its effectiveness as a creative aid in structured ideation processes. Although these further studies could provide more insight, acquiring useful interaction metrics may be much less feasible given more complex studies (like the suggested longitudinal study or using alternative problem statements).

Chapter 5

Conclusions and Further Work

5.1 Ideator Platform Integration of Recommendation System

The Ideator platform utilizes its own API structure (in a repository called IdeatorAPI) so that the platform can make a variety of seamless calls to authenticate users, run supermind ideator moves, chain those moves, and so on. Integrating the recommender system into this structure is pivotal in enhancing user interaction with the platform and potentially nudging users in a helpful direction during their ideation process. In Appendix B, there is an illustration of the necessary code changes needed to employ the recommender system within the API structure. The recommender code uses a state machine approach in Python, where different states represent potential next steps in the ideation process, such as zooming in on details or expanding the scope of consideration. Furthermore, Appendix B provides an overview of how inference on the problem state "scope" is served using Python code and the pre-trained KMeans and embedding models. This inference is passed to the recommender system, along with the prior moves and immediate prior problem statement the user has generated, so that a recommendation for what subsequent move should be taken can be given to the user. This recommendation is manifested visually in the platform through a recommendation button, which appears after every idea generated when a user runs a Supermind Design move (as seen in figure 5.1 below).

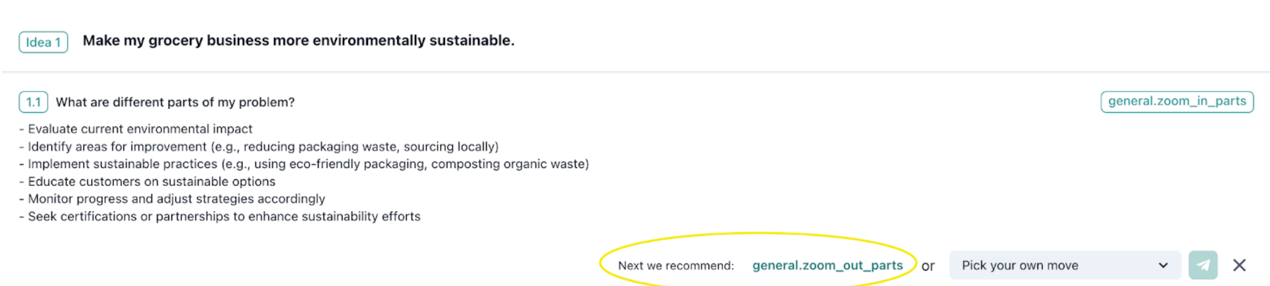

Figure 5.1: Integration of recommendation system in Ideator Platform.

5.2 Ideation Recommendation System Advantages

The ideation "next move" recommendation system offers significant advantages as a powerful tool for enhancing creativity and structured ideation processes. It systematically guides users through complex problem-solving scenarios, making it particularly valuable in contexts that require deep analysis and iterative refinement.

Although the recommendation brings inherent scaffolding to the ideation process, it has the potential to foster innovation. By suggesting thoughtfully curated moves, the system encourages users to explore diverse perspectives and alternative approaches, potentially leading to more creative and comprehensive solutions. Its capacity to adapt based on the user's input problem “scope” enhances its effectiveness, aligning suggestions more closely with where individuals are in the ideation process.

Furthermore, the system's design and integration in the Ideator platform allows for continuous improvement through structured experiments and user feedback. This adaptability is crucial for maintaining relevance and efficacy in rapidly changing environments, making the system a valuable asset for individuals and organizations aiming to enhance their problem-solving capabilities and success.

5.3 Future Works

5.3.1 Dynamic RNN-based Recommendation System

Although a static rules-based recommendation system, informed by machine-learned problem “scope” labeling, is a useful tool to help users progress through the ideation process, there is room to improve upon this design. One approach is to utilize a Recurrent Neural Network model to classify beneficial idea “sequences” (chains of Supermind Ideator moves) to then be able to suggest moves given the current context of a user's idea session.

RNNs, especially LSTMs and GRUs, are proficient in handling variable-length sequences, making them well-suited for the dynamic and iterative nature of the ideation process. They can effectively capture the progression of a user's thought process and adapt “next move” recommendations accordingly. However, it is known that traditional RNNs face challenges in processing very long sequences due to issues of vanishing or exploding gradients. While LSTMs and GRUs mitigate this to some extent, they can still struggle with extremely long sequence dependencies, which might be a limitation in more complex ideation sessions that incorporate much more text over a long period of time. The intuition behind utilizing an RNN is that the typical use case on the Ideator platform does not contain incredibly long sequences of inputs, so we will rarely have to deal with vanishing or exploding gradients.

One major issue with utilizing RNNs is that the inherent subjectiveness of ideation sessions and sequences is too complex to quantitatively label. Thus, without informative labels for idea sequences, it is impossible to utilize the RNNs supervised learning approach, but the study mentioned in section 4 provides a step in the right direction to gain meaningful sequence labels. Moreover, raw text information that is being gathered for each “idea” (user problem input, LLM response after using Supermind Ideator move) after being embedded utilizing well-known text embedding packages (like Doc2Vec) is hard to pull useful patterns from. The following two sections address the embedding issue with potential solutions.

5.3.2 Variational Autoencoder for "Idea" Encoding

This section explores the application of a Variational Autoencoder (VAE) to refine the encoding of "ideas" generated (input problem, output statement, and used move) within the Ideator platform. The VAE extends the capabilities of the traditional autoencoder by introducing a probabilistic graphical model approach where the encoder not only maps the input to a latent representation but also learns the parameters of a probability distribution representing the data.

Approach and Viability The primary advantage of a VAE lies in its ability not just to compress data into a latent space but to model this space as a probability distribution. This characteristic is invaluable in the context of ideation, where each input—comprising a user's problem statement, the AI-generated response, and the ideation move employed—reflects a unique creative thought process. By learning to represent these inputs as distributions, the VAE facilitates a richer understanding of the data's underlying structure, which can then be utilized by an RNN to derive more meaningful features.

In practical terms, the encoder component of the VAE compresses the ideation session data into a latent space while learning the parameters (mean and variance) that define this space's probability distribution. This probabilistic encoding captures the nuances and variability of creative ideas more effectively than a deterministic approach, accommodating the inherent unpredictability and diversity of creative outputs.

The decoder part of the VAE then reconstructs the input data from the probabilistic latent space. This step not only ensures that the space is meaningful but can also be utilized to generate new ideation sequences that are both diverse and coherent with the original inputs. The ability to sample and generate novel ideation sequences from this space directly could prove to be another useful feature in fostering creativity/innovation.

Training the VAE on user ideation "sessions," we aim to enhance the platform's capacity to suggest novel and contextually relevant ideation moves by providing the RNN with more useful features.

In summary, the encoded distributions provided by the VAE serve as enriched inputs for the recommendation system. By utilizing these detailed and nuanced representations, the Dynamic RNN-based Recommendation System can better understand and predict effective next steps in the ideation sequence. The RNN can leverage this rich feature set to offer more accurate and contextually appropriate ideation moves, which can adapt dynamically to the evolving context of a user's session.

Appendix A

Feature Descriptions and Rule Set

A.1 Feature Descriptions

- runType: Preset grouping of Supermind Design moves labeled as either "Explore Problem" or "Explore Solution," and the option to choose other moves with "More Choices."
- __typename: Label of what the data point represents (typically "Response" to relay that the datapoint contains a response from the Ideator API).
- model: Type of fine-tuned LLM used to generate a response to a user's problem.
- bookmarked: Boolean of whether a user bookmarked a data point.
- createdAt: Timestamp of when the move was run.
- temperature: Corresponding temperature used when running the LLM query.
- ideaFrame: Framing of input to LLM given the corresponding Supermind Design move.
- move: The Supermind Design move used when running LLM query.
- problem: Text representing the user's input problem statement.
- updatedAt: Time when the page is updated with response from LLM given corresponding move.
- response: Text response from the LLM given move and problem.
- owner: User ID for datapoint.
- technify: List of technologies utilized if the technify move has been run.
- id: Unique idea for each data point created.
- preference: Categorical information on whether users preferred an output (i.e. 'neutral', 'liked', 'disliked')

- groupify: Type of system utilized when running groupify move (i.e. 'Democracy', 'Market', 'Community', 'Ecosystem', 'Hierarchy', or nan if groupify move not run)
- cognify: Type of terminology used when running cognify move (i.e. 'Creates', 'Learn', 'Decides', 'Senses', 'Decide', 'Learns', 'Remembers', 'Create', or nan if cognify move not run)
- basic: Indication if a "basic" Supermind Design move is run (i.e. 'Zoom Out', 'Zoom In', 'Analogize')
- explorationType: What exploration statement corresponds to the given move (i.e. 'What Am I Missing', 'Better Problem Statement Reformulation', 'Problem Statement Parts')
- solutionType: Solution framing when a unique explore solution move is run (i.e. 'Creative Matrix')
- userGroup: Categorical information to determine where a user signed up from (i.e. an internal study, the main Ideator webpage, etc.)

A.2 Explanation of Rule Set

1. If Zoomed Out (either in "Parts" or "Types"):

   - For specific problem scope:

     - Consider Zooming In (to either "Parts" or "Types") for a more detailed analysis tailored to the specific aspects or types of the problem.

   - For general problem scope:

     - Analogize for creative inspiration that draws from broader or more diverse contexts.

     - Alternatively, explore new moves like Better Problem Statement Reformulation, SDL Search, or What Am I Missing to deepen or broaden understanding before making further moves.

2. If Zoomed In (either in "Parts" or "Types"):

   - For general problem scope:

     - Consider Zooming Out to connect back to broader themes or overarching types, integrating the detailed insights gained.

   - For specific problem scope:

     - Transition into Supermind moves such as Groupify to begin planning execution focusing on collaborative efforts, or Cognify and Technify to refine and enhance solutions through cognitive processes or technological implementations.

3. If Analogized:

   - For specific problem scope:

     - Zoom In on the specifics of the analogous idea to understand detailed correlations or insights.

   - For general problem scope:

     - Initiate Supermind moves to begin planning and execution, choosing strategies based on the analogized concept's overarching needs and context.

4. If Groupified:

   - For general problem scope:

     - Utilize Cognify to enhance the group's decision-making capabilities.

   - For specific problem scope:

     - Technify to provide technological tools that aid the group's tasks.

5. If Cognified:

   - For general problem scope:

     - Consider moving to Groupify identify how to form collective approaches on the problem.

   - For specific problem scope:

     - Employ Technify to delve deeper into how technology can assist with specific solutions.

6. If Technified:

   - For general problem scope:

     -Return to Groupify or Cognify for refinement and enhancement.

   - For specific problem scope:

     - Evaluate whether further technology (a different type or to regenerate outputs) is needed based on the problem's specifics.

Appendix B

Code Listings

B.1 Ideator API Integration

```
@router.post("/recommend/")
def recommend_move(*, token: str = Depends(utils.retrieve_token),
                   req: ideator_v2_models.IdeatorRequest) -> ideator_v2_models.IdeatorResponse:
    try:
        utils.verify_access(req, token)
    except HTTPException as e:
        raise e
    if req.priorMoves:
        # problem_scope = "specific" if len(req.problem.split()) > 20 else "general"  # V1 prob scope
        problem_scope = problemScopeNLP.nlp.InferProblemScope.infer_query_label(req.problem)[0] # V2 prob scope using ML
        curr_state = recommender.get_state_from_move_path(req.priorMoves[-1])
        recommended_move = recommender.detailed_sequential_logic(current_state=curr_state, problem_scope=problem_scope)
        return ideator_v2_models.IdeatorResponse(
            request={
                "priorMoves": req.priorMoves,
                "problem": req.problem
            },
            response={"nextMove": recommended_move}
        )
    else:
        raise utils.HTTPException(status_code=400, detail="priorMoves empty")
```

B.2 Scope Analysis Inference

```
def download_nltk_data():
    packages = ['stopwords', 'wordnet', 'punkt']
    for package in packages:
        try:
            print(f"Checking if '{package}' is already downloaded..."
                )
            nltk.data.find(f'tokenizers/punkt/{package}.zip')
            print(f"'{package}' is already downloaded.")
        except LookupError:
            nltk.download(package)

### Helper Functions

def clean_and_lemmatize(query, stop_words):
    lemmatizer = WordNetLemmatizer()
    query = re.sub(r'[^\w\s]', '', query) # Removes punctuation
    words = nltk.word_tokenize(query)
    lemmatized = [lemmatizer.lemmatize(word.lower()) for word in
        words if word.lower() not in stop_words]
    return ' '.join(lemmatized)

def is_number(token):
    try:
        float(token.replace(',', ''))
        return True
    except ValueError:
        return False

def syllable_count(word):
    word = word.lower()
    count = 0
    vowels = "aeiouy"
    if word[0] in vowels:
        count += 1
    for index in range(1, len(word)):
        if word[index] in vowels and word[index - 1] not in vowels:
            count += 1
    if word.endswith("e"):
        count -= 1
    if count == 0:
        count += 1
    return count

def label_specific_or_general(df):
    for main_cluster in df['main_cluster'].unique():
```

```
        main_cluster_df = df[df['main_cluster'] == main_cluster]

        if main_cluster_df['is_largest_main_cluster'].any():
            for sub_cluster in main_cluster_df['sub_cluster'].unique():
                # Directly modifying the original DataFrame here
                indices = main_cluster_df[main_cluster_df['sub_cluster'] == sub_cluster].index
                df.loc[indices, 'heuristic_label_name'] = assign_labels_based_on_majority(df.loc[indices])
        else:
            indices = main_cluster_df.index
            df.loc[indices, 'heuristic_label_name'] = assign_labels_based_on_majority(df.loc[indices])

    return df

def assign_labels_based_on_majority(cluster_df):
    """
    Assigns "specific" or "general" labels to a cluster based on the majority/minority status of heuristic clusters.
    """
    # Count the number of queries in each heuristic cluster
    label_counts = cluster_df['heuristic_cluster'].value_counts()

    # Determine minority and majority labels
    minority_label = label_counts.idxmin()
    majority_label = label_counts.idxmax()

    # Assign "specific" to the minority label and "general" to the majority label
    cluster_df['heuristic_label_name'] = cluster_df['heuristic_cluster'].apply(lambda x: 'specific' if x == minority_label else 'general')

    return cluster_df

def is_match(label1, label2):
    return label1 == label2

### Load stored models

def load_stored_models():
    model_keys = ['doc2vec', 'main_clustering', 'sub_clustering', 'heuristic_clustering', 'normalization_parameters', 'largest_cluster_idx', 'heuristic_label_mapping']
    stored_models_in = {}
```

```python
    for key in model_keys:
        with open(f'/Users/danielpapacica/Desktop/ideatorAPI/problemScopeNLP/models/{key}.pkl', 'rb') as file:
            stored_models_in[key] = pickle.load(file)
    return stored_models_in

def load_query_debug_info():
    with open('/Users/danielpapacica/Desktop/ideatorAPI/problemScopeNLP/models/query_debug_info.pkl', 'rb') as file:
        query_debug_info = pickle.load(file)
    return query_debug_info

def load_global_nlp_model():
    with open('/Users/danielpapacica/Desktop/ideatorAPI/problemScopeNLP/models/global_nlp_model.pkl', 'rb') as file:
        global_nlp_model = pickle.load(file)
    return global_nlp_model

def load_global_stop_words():
    with open('/Users/danielpapacica/Desktop/ideatorAPI/problemScopeNLP/models/global_stop_words.pkl', 'rb') as file:
        global_stop_words = pickle.load(file)
    return global_stop_words

### Loaded Global Variables

stored_models_from_save = load_stored_models()
query_debug_info = load_query_debug_info()
global_nlp_model = load_global_nlp_model()
global_stop_words = load_global_stop_words()

global_mean, global_std = 0, 0

### Inference Functions

def preprocess_single_query(query, custom_stopwords={'new', 'use', 'want'}):
    stop_words = global_stop_words
    cleaned_query = clean_and_lemmatize(query, stop_words)
    return cleaned_query

def infer_embedding_2(model, query, epochs=20, alpha=0.025, seed=8):
    model.random.seed(seed)
    embedding = model.infer_vector(query.split(), epochs=epochs, alpha=alpha)
    return embedding

```

```
def calculate_heuristic_2(query, nlp_model, stop_words, input_mean, input_std):
    tokens = word_tokenize(query)
    sentences = sent_tokenize(query)
    total_sentences = len(sentences)
    total_words = len(tokens)
    total_syllables = sum(syllable_count(word) for word in tokens)

    unique_words_count = len(set(tokens))
    avg_sentence_length = total_words / total_sentences if total_sentences > 0 else 0
    lexical_diversity = unique_words_count / total_words if total_words > 0 else 0

    doc = nlp_model(query)
    entity_count = len(doc.ents)
    numerical_data_count = sum(1 for token in tokens if token.isdigit() or token.isnumeric())
    meaningful_words_count = len([token for token in tokens if token.lower() not in stop_words])

    readability_score = 206.835 - 1.015 * (total_words / total_sentences) - 84.6 * (total_syllables / total_words) if total_words > 0 else 0

    # Pack features into an array
    features = np.array([meaningful_words_count, unique_words_count, entity_count, numerical_data_count, avg_sentence_length, lexical_diversity, readability_score])

    # NORMALIZED VERSION
    # Normalize features with safeguard against division by zero
    normalized_features = np.zeros(features.shape)
    for i in range(len(features)):
        if input_std[i] > 0:
            normalized_features[i] = (features[i] - input_mean[i]) / input_std[i]
        else:
            # Handle features with zero std; potentially keep as is, set to 0, or we could use some other logic
            normalized_features[i] = 0

    normalized_features = normalized_features.reshape(1, -1)

    # NON-NORMALIZED VERSION
    # normalized_features = features.reshape(1, -1)

```

```python
    return normalized_features

def infer_query_label(query, models=stored_models_from_save,
    nlp_model=global_nlp_model, stop_words=global_stop_words,
    global_mean=global_mean, global_std=global_std):
    debug_info = {}  # store and return debugging information
    best_random_state = 31
    # Preprocess and clean the query
    cleaned_query = preprocess_single_query(query)
    debug_info['cleaned_query'] = cleaned_query

    # Infer embedding for the query
    embedding = infer_embedding_2(models['doc2vec'], cleaned_query,
        seed=best_random_state)
    debug_info['embedding'] = embedding.tolist()

    # Predict main cluster
    main_cluster_label = models['main_clustering'].predict([embedding
        ])[0]
    debug_info['main_cluster_label'] = main_cluster_label

    # Check if the query belongs to the largest cluster
    if main_cluster_label == models['largest_cluster_idx']:
        # Part of the largest cluster, use the sub_clustering model
            to determine sub-cluster
        sub_cluster_label = models['sub_clustering'].predict([
            embedding])[0]
        cluster_label_name = f'sub_{sub_cluster_label}'
    else:
        # Not part of the largest cluster, use main cluster label
        cluster_label_name = f'main_{main_cluster_label}'
    debug_info['cluster_label_name'] = cluster_label_name

    # Retrieve normalization parameters for the predicted cluster
    normalization_params = models['normalization_parameters'].get(
        cluster_label_name)
    if normalization_params:
        cluster_mean = normalization_params['mean']
        cluster_std = normalization_params['std']
    else:
        # Fallback to global mean and std if no specific parameters
            exist for this cluster
        print("ERROR OCURRED RETRIEVING NORM VALS")
        cluster_mean = global_mean
        cluster_std = global_std

    # Infer heuristics for the query
```

```python
    # heuristic_features = calculate_heuristic_2(cleaned_query, nlp_model, stop_words, cluster_mean, cluster_std) # V1 w/ cleaned query
    heuristic_features = calculate_heuristic_2(query, nlp_model, stop_words, cluster_mean, cluster_std) # V2 w/ not cleaned query (THIS PRODUCES BETTER FINAL MATCH PERCENTAGE w/ Human Labels)
    debug_info['heuristic_features'] = heuristic_features.tolist()

    # Predict heuristic cluster using the correct model based on the cluster_label_name
    heuristic_model = models['heuristic_clustering'].get(cluster_label_name)
    if heuristic_model:
        heuristic_cluster = heuristic_model.predict(heuristic_features)[0]
    else:
        # Fallback if no specific heuristic model exists for this label
        print("Heuristic Model Not Found!")
        heuristic_cluster = 0
    debug_info['heuristic_cluster'] = heuristic_cluster

    # Determine label based on the heuristic cluster
    heuristic_label_mapping = models['heuristic_label_mapping'].get(cluster_label_name)
    if heuristic_label_mapping and isinstance(heuristic_label_mapping, dict):
        specific_clusters = heuristic_label_mapping.get('specific', [])
        if not isinstance(specific_clusters, list):
            specific_clusters = [specific_clusters]
        label = 'specific' if heuristic_cluster in specific_clusters else 'general'
    else:
        label = heuristic_label_mapping  # Direct mapping if only one label type exists
    debug_info['inferred_label'] = label

    return label, debug_info
```

References

[1] S. R. Rick, G. Giacomelli, H. Wen, R. J. Laubacher, N. Taubenslag, J. L. Heyman, and T. W. Malone, *Supermind ideator: Exploring generative ai to support creative problem-solving*, 2023. arXiv: 2311.01937 [cs.AI]. URL: http://arxiv.org/abs/2311.01937.

[2] A. Cai, S. R. Rick, J. L. Heyman, Y. Zhang, A. Filipowicz, M. Hong, and T. Malone, *Designaid: Using generative ai and semantic diversity for design inspiration*, 2023. DOI: 10.1145/3582269.3615596.

[3] A. Campero, M. Vaccaro, J. Song, H. Wen, A. Almaatouq, and T. W. Malone, *A test for evaluating performance in human-computer systems*, arXiv preprint, 2022. arXiv: 2206.12390 [cs.HC]. URL: http://arxiv.org/abs/2206.12390.

[4] A. Koppineni, D. Kong, and T. Malone, *Supermind design for responding to covid-19: A case study of university students generating innovative ideas for a societal problem*, Feb. 2022. URL: https://cci.mit.edu/wp-content/uploads/2022/03/2022_WP_Koppineni_Supermind.pdf.

[5] P. Osterman, "How american adults obtain work skills: Results of a new national survey," *ILR Review*, vol. 75, no. 3, pp. 578–607, 2022. DOI: 10.1177/00197939211018191.

[6] M. el Haiba, L. EL BASSITI, and R. Ajhoun, "Using recommender systems to support idea generation stage," *Journal of Fundamental and Applied Sciences*, vol. 9, 2017. DOI: 10.4314/jfas.v9i7s.122.

[7] M. C. for Collective Intelligence, *Supermind design primer*, 2021. URL: https://cci.mit.edu/wp-content/uploads/2021/08/Supermind-Design-Primer-v2.pdf.

[8] D. Council. "Framework for innovation," Design Council. (2024), URL: https://www.designcouncil.org.uk/our-resources/framework-for-innovation/ (visited on 05/01/2024).

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, https://www2.cs.uh.edu/~ceick/7363/Papers/dbscan.pdf, Presented at the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), 1996.

[10] V. Cohen-Addad, V. Kanade, F. Mallmann-Trenn, and C. Mathieu, *Hierarchical clustering: Objective functions and algorithms*, 2017. arXiv: 1704.02147 [cs.DS].

[11] T. Weißer, T. Saßmannshausen, D. Ohrndorf, P. Burggräf, and J. Wagner, "A clustering approach for topic filtering within systematic literature reviews," *MethodsX*, vol. 7, p. 100 831, 2020, PMID: 32195145, PMCID: PMC7078380. DOI: 10.1016/j.mex.2020.100831. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7078380/.

[12] Q. V. Le and T. Mikolov, *Distributed representations of sentences and documents*, 2014. arXiv: 1405.4053 [cs.CL].

[13] "Tf–idf," in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Boston, MA: Springer US, 2010, pp. 986–987, ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_832. URL: https://doi.org/10.1007/978-0-387-30164-8_832.

[14] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, 2013. arXiv: 1301.3781 [cs.CL].

[15] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. URL: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf.

[16] F. L. Gewers, G. R. Ferreira, H. F. D. Arruda, F. N. Silva, C. H. Comin, D. R. Amancio, and L. D. F. Costa, "Principal component analysis: A natural approach to data exploration," *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–34, May 2021, ISSN: 1557-7341. DOI: 10.1145/3447755. URL: http://dx.doi.org/10.1145/3447755.

[17] X. Schmitt, S. Kubler, J. Robert, M. Papadakis, and Y. LeTraon, "A replicable comparison study of ner software: Stanfordnlp, nltk, opennlp, spacy, gate," in *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, 2019, pp. 338–343. DOI: 10.1109/SNAMS.2019.8931850.

[18] M. Solnyshkina, R. Zamaletdinov, L. Gorodetskaya, and A. Gabitov, "Evaluating text complexity and flesch-kincaid grade level," *Journal of Social Studies Education Research*, vol. 8, pp. 238–248, Nov. 2017.

[19] J. Treffers-Daller, P. Parslow, and S. Williams, "Back to basics: How measures of lexical diversity can help discriminate between cefr levels," *Applied Linguistics*, vol. 39, pp. 302–327, 3 Jun. 2018. DOI: 10.1093/applin/amw009. URL: https://doi.org/10.1093/applin/amw009.

[20] Y. Zhang and W. Wu, "How effective are lexical richness measures for differentiations of vocabulary proficiency? a comprehensive examination with clustering analysis," *Language Testing in Asia*, vol. 11, no. 15, 2021. DOI: 10.1186/s40468-021-00133-6. URL: https://doi.org/10.1186/s40468-021-00133-6.

[21] Z. Susoy, "Lexical density, lexical diversity and academic vocabulary use: Differences in dissertation abstracts," *Journal of Education and Learning*, vol. 8, no. 2, pp. 198–207, 2023. URL: https://files.eric.ed.gov/fulltext/EJ1406507.pdf.

[22] Y. Bestgen, *Measuring lexical diversity in texts: The twofold length problem*, 2023. arXiv: 2307.04626 [cs.CL].

[23] N. Bérubé, M. Sainte-Marie, P. Mongeon, and V. Larivière, "Words by the tail: Assessing lexical diversity in scholarly titles using frequency-rank distribution tail fits," *PLoS One*, vol. 13, no. 7, e0197775, Jul. 2018. DOI: 10.1371/journal.pone.0197775.

[24] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation," *Electronics*, vol. 9, no. 8, 2020, ISSN: 2079-9292. DOI: 10.3390/electronics9081295. URL: https://www.mdpi.com/2079-9292/9/8/1295.