FROM NATURAL LANGUAGE TO CLEAR VISUALS: CREATING SMART VISUALIZATIONS WITH RETRIEVAL-AUGMENTED GENERATION TO EMPOWER MATHEMATICAL LEARNING

Raeanne Li [1], Carlos Gonzalez [2]

[1] The Harker School, 500 Saratoga Ave, San Jose, CA 95129
[2] University of California, Berkeley, CA 94720

ABSTRACT

For many students, mathematics is a challenging subject because of its abstract nature. Through my own learning experience, I realized how visual representations can simplify complex problems, making them intuitive and engaging. These moments sparked my passion for exploring how visualization can empower math learners, especially those with diverse learning needs and styles, to overcome the barriers associated with traditional teaching methods [1]. Students with dyslexia, ADHD, or autism often face challenges with text-heavy explanations and abstract concepts, but visualization tools, such as charts, diagrams, and interactive models, can make a difference, offering alternative approaches that better support their learning [2].

This project focuses on developing an AI-powered visualization platform designed to generate visual representations for math word problems. With Retrieval-Augmented Generation (RAG), the smart system retrieves relevant data from external sources and generates content-specific math problems, ensuring high accuracy and alignment with user queries. A key contribution of this research is the integration of a dual-LLM architecture with RAG to enhance diagram creation. The first LLM generates clear, concise, and imperative instructions from natural language queries, while the second LLM translates these instructions into valid Scalable Vector Graphics (SVG) code for precise diagrams. The integrated approach allows for automated, scalable, and customizable diagram generation, offering an engaging and accessible learning experience for different problems. Ultimately, the smart system combines problem generation and visualization into a unified web and mobile application, providing diverse learners with powerful tools to engage with math.

KEYWORDS

Math Visualization, Accessible Learning, Dual-LLM Architecture, AI-Powered Learning Tools

David C. Wyld et al. (Eds): SEC, DMA, AIAPP, CYBI, CoSIT, SIGL, NLPML – 2025
pp. 79-90, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150806

1. INTRODUCTION

1.1. Motivation

Many young students find it challenging to understand math concepts quickly. The challenge is amplified when dealing with word problems [3]. Visualization simplifies complex problems by breaking them into smaller, manageable parts, making them easier to comprehend. Tools such as diagrams, flowcharts, and color-coded elements help students retain formulas, understand problem-solving steps, and recognize relationships between variables. By visualizing functions, equations, shapes, and angles, abstract concepts become clearer and more accessible [4].

The benefits of visualization tools are not limited to conventional learners but encompass all types of students and learning styles. Students with dyslexia often struggle with word problems and symbolic representations, and visualization can help reduce their dependence on text [5]. It also benefits students with ADHD who have a tough time focusing, especially during mentally arduous tasks, such as learning abstract math concepts [6] [7]. Using graphics and visualizations can make math lessons more engaging and stimulating. Diagrams and charts are also especially helpful for students with Autism Spectrum Disorder (ASD), who are strong visual thinkers [8] [9]. Visual strategies are effective in reducing their anxiety, offering them a structured way to understand patterns, shapes, and problem-solving processes.

Visualization tools are valuable in supporting students with diverse learning needs and styles by providing alternative ways for them to engage with math. They offer creative and adaptable approaches to teach math concepts, making math accessible and understandable for all.

1.2. Objectives

Recent advancements in Artificial Intelligence (AI) have made content generation easier than ever, opening up many new possibilities [10]. Generative AI tools such as OpenAI's DALL-E can convert written descriptions into high-quality images, making extensive expertise no longer needed for creating incredible visual products. AI-generated visuals are already making significant impacts in industries like film, fashion, and even medical training. However, their potential in education is underexplored, presenting exciting opportunities to enhance learning experiences.

This project aims to provide an inclusive learning approach by integrating visualization strategies into word problem-solving, a crucial yet challenging aspect of math education. By developing an AI-powered platform, I seek to generate various types of visual representations, such as tape diagrams and Venn diagrams, that transform complex problems into clear and intuitive illustrations. Many students struggle with text-based methods, finding it difficult to interpret abstract concepts without a visual reference. This tool is designed to serve as a fast, reliable, and adaptable learning aid, making math more accessible for students with diverse learning needs. By redefining the way educational content is created, this innovation not only enhances comprehension but also empowers educators with advanced tools to efficiently generate dynamic learning materials, tailor instruction to diverse student needs, and present information in a more engaging and impactful way. Ultimately, this project envisions a future where AI revolutionizes the way knowledge is conveyed and absorbed in the classroom and beyond.

2. METHOD

2.1. Conceptual Framework

The Retrieval-Augmented Generation (RAG) model is a conceptual framework designed for LLMs and generative AI [11]. By combining retrieval and generation processes, RAG enhances the effectiveness and relevance of AI-generated content [12]. The model first retrieves relevant information from multiple external sources, such as databases or search engines, based on the input query [13]. Then, it uses a generative model, such as GPT-4, to process that information and turn it into coherent and contextual responses. The integration makes RAG extremely powerful for tasks requiring deep knowledge, such as answering open-ended questions, summarizing long documents, or making interactive dialogues [14]. The diagram below (Figure 1) illustrates the RAG model and how it references outside material to produce better responses [15]. This technology could have a significant impact on how we support and advance complex educational efforts.

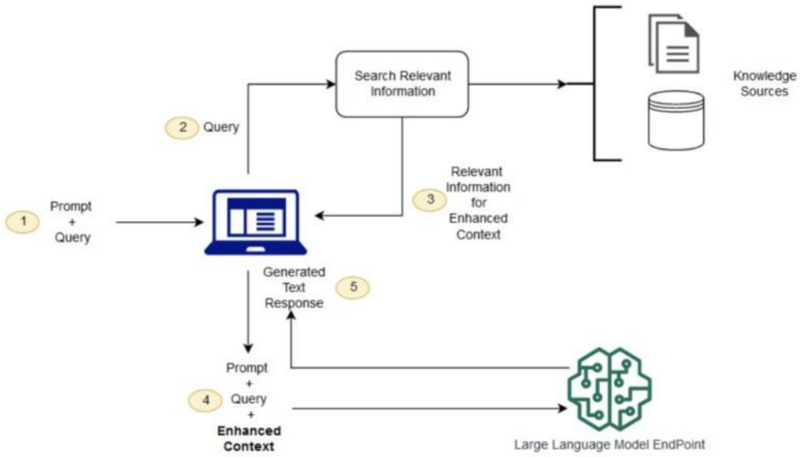


Figure 1. Retrieval-Augmented Generation Model

Tools like DALL-E generate images from text by encoding the description into a latent representation, capturing the core concepts described in the prompt. It starts with a noisy initial image, which the diffusion model iteratively refines by reducing noise, guided by a loss function. Trained on a vast dataset of text-image pairs, DALL-E learns to create realistic and creative images across various styles, aligning with the input text [16]. The ability to convert text into images allows users to create visual content without needing special training. Generative visualization tools make personalized image creation accessible and scalable.

The intended system utilizes a Retrieval-Augmented Generation (RAG) framework, enabling it to retrieve relevant data from external sources and generate content-specific math problems with high accuracy (Figure 2). A key feature of the system is the integration of a dual-LLM architecture with RAG to enhance diagram creation. The first LLM interprets natural language queries and generates clear, concise, and actionable instructions, while the second LLM converts these instructions into Scalable Vector Graphics (SVG) code, resulting in precise, high-quality diagrams.

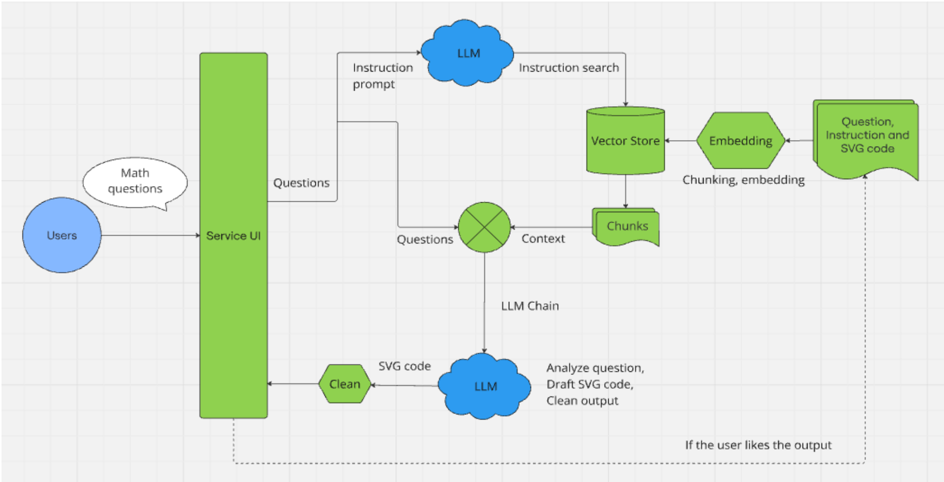


Figure 2. Framework of the Smart Visualization System

2.2. Research Design

I organized my work by breaking the project into its core components and approaching each part methodically, while allowing flexibility for trial and unexpected challenges. The project consists of three main components: problem understanding with OpenAI's LLMs, diagram generation with the RAG model, and the development of an online application that integrates the two components to make the resources accessible to the community. Each component plays a crucial role in ensuring that the resulting application will be functional and effective for a diverse range of learners.

Problem Understanding. For the problem understanding component, or essentially the more "human"-like part of the whole task, I decided to use an LLM to grab the most important parts of the question and leave out all the irrelevant surrounding words, in order to create clear, concise, and imperative step-by-step instructions for producing SVG code, to be followed by a separate function later on. I gave as example contexts manually created sets of instructions that yielded good results when OpenAI's GPT-4o was then asked to create valid SVG code based on them.

Diagram Generation. I then created a CSV of the example questions, SVG instructions, and code that were successful. This data was integrated with a Retrieval-Augmented Generation (RAG) model to generate appropriate SVG code according to instructions given by the initial LLM process. The RAG model allows the system to retrieve similar data from a vector store and enhance the original input with relevant information. The use of RAG eliminates the need to retrain a large language model while ensuring that the generated responses are accurate and aligned with the underlying mathematical goals.

Initially, I considered using DALL-E for this purpose, but its outputs often geared toward artistic interpretations of the problem than logic-driven diagrams for mathematical explanations, I also explored other tools like Wolfram Alpha for its advanced graphics capabilities, HTML for web integration, and QuickChart for generating high-quality charts. After trying all of them, I decided to stick with not using another API because these largely never served the correct purpose or even generated accurate diagrams.

As a result, I shifted to using OpenAI's LLMs to "understand" the problems and generate precise instructions for creating SVG diagrams from scratch. After several iterations, this approach produced the best outcomes. I was able to create problem-specific diagrams that were not overrendered visuals or nonsensical charts, but instead clear and accurate diagrams better aligned with the math problems.

Online Application. The final component involves developing a web/mobile application that integrates the math problems with their corresponding visual diagrams. This application will provide diverse learners with personalized access to mathematical visualizations, helping them better understand abstract concepts. The question-asking interface, or the “ask page” of the application, takes in user’s questions and returns a brief solution along with the generated diagram. With the core components in place, I integrated them into a cohesive system using Android Studio, connecting the backend logic with the front-end interface.

3. PROCESS

3.1. Prototype Development

The UI application has been set up to demonstrate key features of the desired visualization platform. The prototype uses a clean layout to illustrate the integration of key components including GPT-4 for content generation, Flutter for cross-platform UI development, and Firebase for backend services.

After logging in, users arrive at a main page (Figure 3) that directs them to an “Ask Page” or a “Learn Page”. In the “Ask” interface (Figure 4), users can type in a math problem and interact with the chatbot to get corresponding diagrams that support their learning. Within the “Learn” page (Figure 5), users are provided a list of math problems, each paired with accurate visuals to illustrate math concepts and enhance understanding.

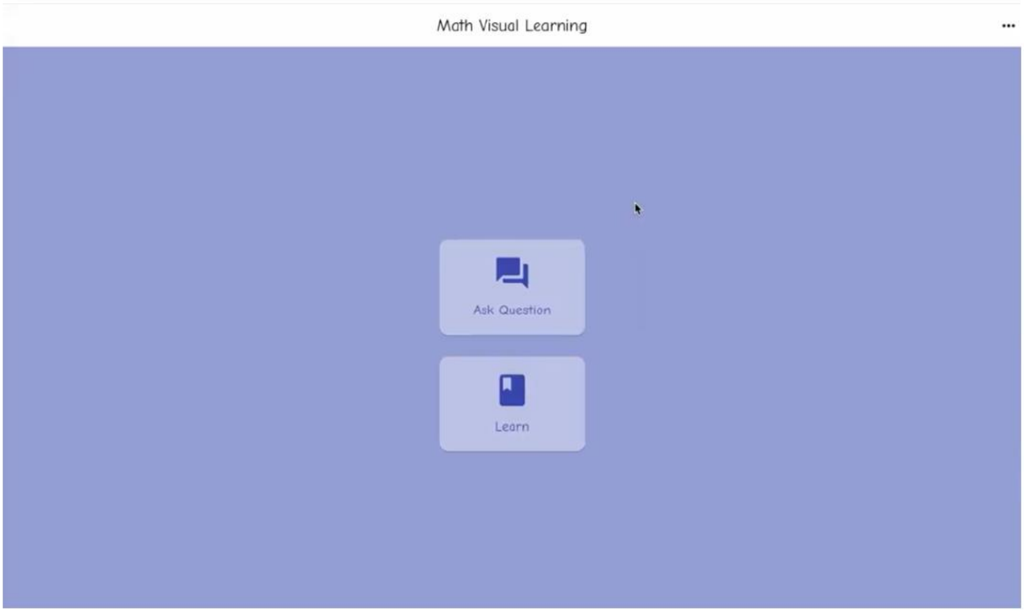


Figure 3. Main Page

Figure 4. “Ask Page” Chatbot

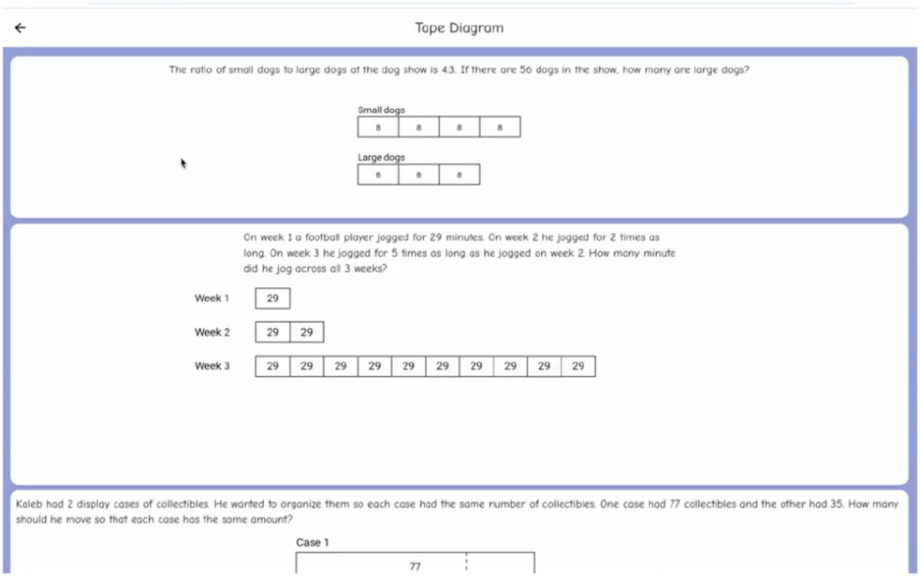


Figure 5. Learn Page

By combining these features and components, the initial implementation serves as proof that these components can work cohesively to deliver a solution that is flexible, reliable, and scalable.

3.2. Attempts

The main challenge I had was to generate accurate diagrams that illustrate the logic behind the word problems. I have explored several tools and methods, but each has presented limitations that prevent the creation of precise diagrams.

DALL-E: My first attempt was to use DALL-E, which is capable of generating visually appealing images. However, the outputs reveal that it is more geared toward artistic interpretations of the problem than logic-driven diagrams for mathematical explanations (Figure 6). To address this, I manually created diagrams and used ChatGPT to generate a description. I then fed it back into DALL-E, hoping it would refine the output. Unfortunately, this approach also failed, as DALL-E continued to generate representations without logical clarity. There is also a persistent flaw when it comes to forming coherent letters and numbers (Figure 7).

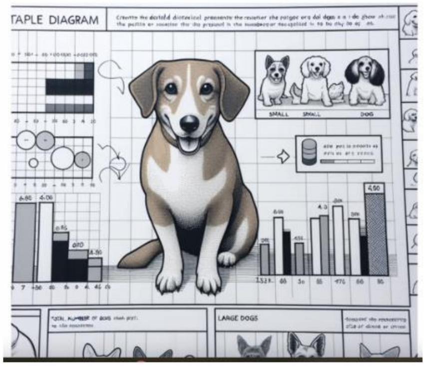


Figure 6. Example Interpretation by DALL-E

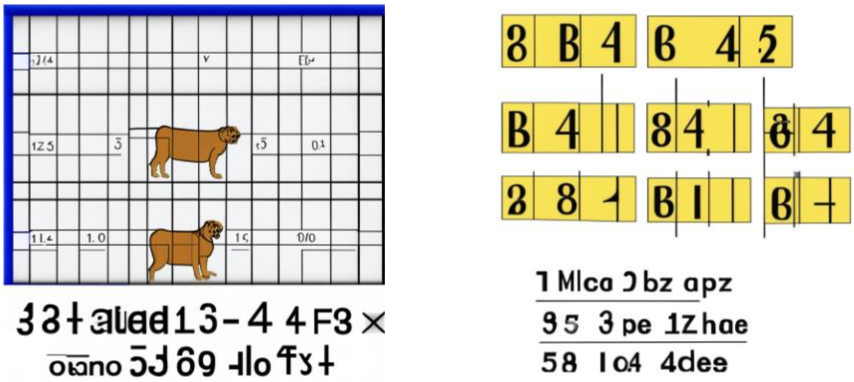


Figure 7. Example Letters and Numbers by DALL-E

Wolfram Alpha: To address the challenge of generating clear and logical diagrams, I then turned to Wolfram Alpha. The primary challenge however was running Wolfram Alpha within the Python environment. The implementation was too complex for my level of expertise. Despite exploring online solutions for API integration, I could not incorporate its diagramming capabilities into the workflow.

HTML and Draw.io: After Wolfram Alpha, I decided to try generating HTML. The intention was to use a more integrated approach within the web development environment. ChatGPT was able to generate functional HTML code for simple diagrams, but the outputs were of low quality (Figure 8). I also tried using Draw.io to create diagrams and export them as HTML, but while the results were better, as the examples increased and became more complex, ChatGPT struggled to produce complete HTML code. This made it challenging to scale the approach for the diverse diagram requirements.

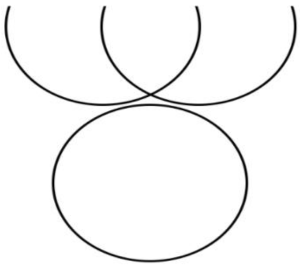


Figure 8. Example Diagram using HTML

QuickChart: I also tried Quickchart, an open-source tool good for generating charts and graphs. However, the generated charts often failed to align with the underlying mathematical logic. For example, when trying to create a visual to explain the relationship between a circle's radius and circumference, QuickChart created a pie chart that misrepresented the radius as a segment of a whole (Figure 9). Such mismatches limited its capability to produce precise visualizations.

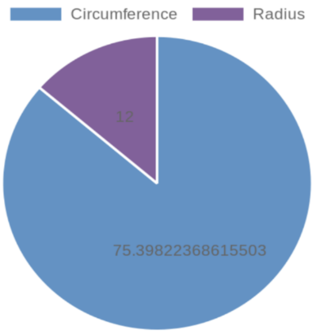


Figure 9. Example Diagram by QuickChart

4. RESULTS

Despite earlier challenges with various tools, significant progress was made in generating the diagrams for mathematical problems. The final solution, using Scalable Vector Graphics (SVG) code instead of HTML, successfully generated more accurate and logical diagrams for complex mathematical concepts, such as Venn diagrams and tape diagrams using geometric shapes. These outcomes highlight the potential of utilizing AI to make visually clear and semantically accurate representations of word problems.

4.1. Solution

The final solution leverages the RAG model to efficiently generate SVG diagrams from natural language descriptions. This approach integrates advanced semantic retrieval and language model capabilities, significantly improving the accuracy and clarity of the generated visuals. The workflow is summarized as follows:

First, a large dataset of math problems, solutions, and descriptive diagrams from Hugging Face was used. This dataset contained both textual and visual content, which was the foundation for the retriever's searchable documents. To enable high-speed semantic search, text data was embedded using the advanced sentence-transformers/all-mpnet-base-v2 model and stored in a Chroma vector database.

Then, natural language queries describing the desired diagrams were processed to retrieve contextually relevant information from the vector database based on semantic similarity. This ensured that generated diagrams aligned precisely with the problem requirements.

After this stage was processed, a secondary GPT-4 model was used to generate those clear LLM-readable instructions from the original user query, or the word problem. These problems may be AI-generated, but manually created problems work here as well. The output instructions, formatted imperatively, provided a primary LLM with the directions for SVG generation, improving the fidelity and usability of the resulting diagrams.

Short, clear, and imperative commands were found to be most effective for generating diagrams. Using OpenAI's GPT-4, the smart visualization system produced well-structured SVG code for diagrams. The results were significantly cleaner and more efficient compared to the previously attempted HTML/draw.io-based approach. A carefully designed prompt template ensured that the language model consistently generated accurate and visually clear outputs.

4.2. Outcomes

The evaluation of the system's performance highlighted its effectiveness in generating accurate and clear diagrams across various mathematical contexts. In particular, the smart system performed well in creating tape diagrams and Venn diagrams, two common tools used to simplify complex concepts in mathematics.

The smart system successfully generated tape diagrams to represent proportional reasoning, a key concept in areas like ratios and fractions. These diagrams effectively illustrated relationships between parts and wholes, helping to make abstract ideas more concrete. The clear, visual nature of the diagrams was especially useful for students who may struggle with more traditional methods of explanation, providing a straightforward way to grasp proportional relationships (Figure 10).

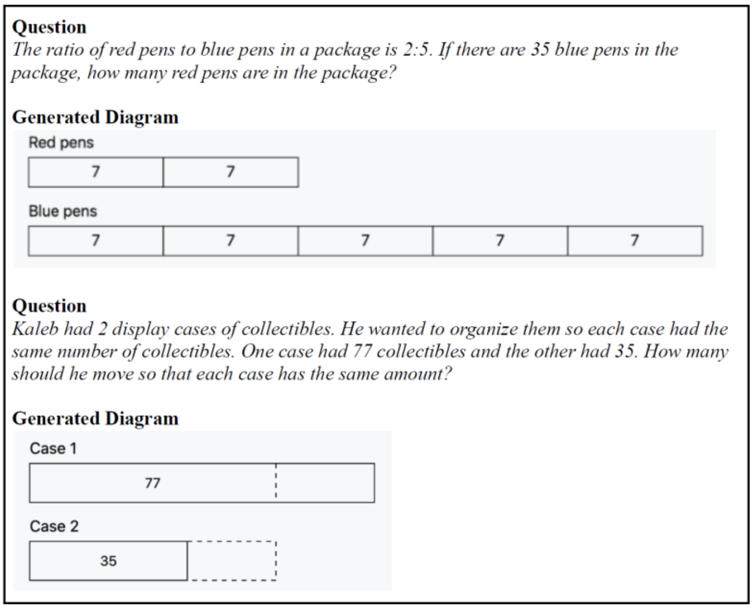


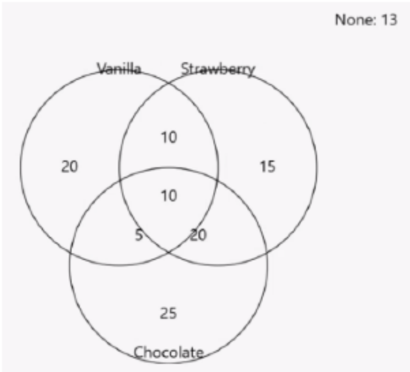
Figure 10. Example Tape Diagrams

The system also produced effective Venn diagrams, which are crucial for illustrating relationships between sets in topics like set theory, logic, and probability. The diagrams clearly displayed intersections, unions, and differences between sets, making complex set-based concepts easier to understand. This functionality demonstrated the system's ability to handle a wide range of mathematical topics, supporting students in comprehending abstract ideas like set operations and logical relationships (Figure 11).

Questions

In a survey among 118 students, 45 like vanilla flavored ice cream, 55 like strawberry flavored ice cream, 60 like chocolate flavored ice cream, 20 like vanilla and strawberry, 15 like vanilla and chocolate, 30 like strawberry and chocolate, and 10 like all three flavors. Find: a) students who don't like any of the three flavors b) students who like vanilla and strawberry, but not chocolate.

Generated Diagram



Questions

In a survey among 170 students, 80 liked to play soccer, 90 liked to play basketball, 100 liked to play tennis, 40 liked soccer and tennis, 30 liked to play soccer and basketball, 50 liked tennis and basketball, and 15 liked to play all types of games. Find: a) students who play only soccer b) students who play tennis and basketball but not soccer.

Generated Diagram

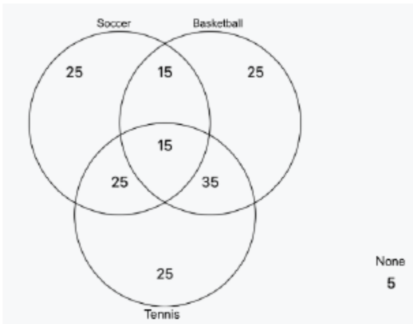


Figure 11. Example Venn Diagrams

5. CONCLUSIONS

This research addressed the challenge of generating mathematical conceptual diagrams for educational applications, focusing on precision rather than photorealistic or artistic renderings. It demonstrated that traditional approaches, such as What You See Is What You Get (WYSIWYG) diagram editors, might not always be the most efficient or scalable for creating mathematical diagrams. Rather than relying on manual, drag-and-drop interfaces or artistic tools like DALL-E (which, though not strictly WYSIWYG, emphasizes pixel-level image generation), this study introduced a novel approach that uses a code-first methodology based on SVG generation. By focusing on imperative instructions instead of direct pixel manipulation, the system constructs diagrams that prioritize logical structure and mathematical accuracy. This method enables the creation of clear, precise visualizations that are well-suited to academic contexts.

The key difference in this approach is its emphasis on semantic accuracy and reusability, rather than aesthetic appeal. SVG-based outputs are scalable, editable, and tailored to the logical needs of academic settings, making them ideal for teaching and research. By separating the logic of the diagram from pixel-level rendering, this method lays the foundation for a reproducible, transparent process for generating mathematical visualizations.

A major innovation in this research was the integration of a Retrieval-Augmented Generation (RAG) framework. This framework allowed the system to efficiently generate diagrams for a wide variety of mathematical problems. By embedding textual data with the sentence-transformers/all-mpnet-base-v2 model and storing it in a Chroma vector database, the system achieved high-speed semantic retrieval. This ensured that the generated diagrams closely aligned with user queries, enhancing accuracy and relevance. This RAG-driven architecture not only addresses scalability by supporting a broad range of mathematical concepts but also maintains high precision. Its flexibility makes it suitable for educational applications, where accuracy is critical, as even minor mistakes can affect learning outcomes.

A key innovation of this research is the introduction of a dual-LLM architecture designed to enhance usability and reliability in diagram generation. In this framework, the first language model transforms natural language user queries into clear, LLM-optimized instructions, while the second model interprets these instructions and generates valid SVG code. By separating the roles of interpretation and generation, the system maintains logical consistency while ensuring the accuracy of the resulting visual output.

This code-centric approach offers distinct advantages over traditional WYSIWYG (What You See Is What You Get) tools. Rather than relying on manual element manipulation, the system leverages the precision and flexibility of SVG code generation, enabling automation and reducing user effort. Users can simply describe their visualization needs in natural language, and the system autonomously renders the appropriate diagram—bridging the gap between intuitive intent and executable design.

This approach also distinguishes itself from models like DALL-E, which often generate artistic or photorealistic images that lack the logical clarity needed for academic diagrams. While DALL-E and similar tools can produce visually compelling outputs, their focus on pixel-level rendering can lead to inaccuracies, particularly in text, numbers, and structural relationships. In contrast, the SVG-based method adopted in this research ensures that the generated diagrams are logically grounded and well-suited for educational use, offering a more reliable solution for generating precise, contextually appropriate diagrams.

This research has significant potential for replication and expansion. The code-first approach to diagram generation lays a strong foundation for further developments in AI-driven educational tools. Within mathematics alone, re-engineering the LLM prompts and altering the example data that the RAG model can access to a new topic can help produce diagrams belonging to even more categories, like trigonometry diagrams or systems of equations representations. The approach could be applied to other subjects as well, including chemistry word problems or physics free body diagrams.

By bridging a gap between AI-driven automation and academic rigor, this project demonstrates how technology can be harnessed to improve educational outcomes and knowledge dissemination. It combines artificial intelligence, visualization, and a platform to create a tool that benefits all students wanting to learn, which could inspire more efforts to leverage new technologies for the educational scene rather than limiting student contact with it.

REFERENCES

[1] Ajogbeje, Oke James. "Enhancing classroom learning outcomes: The power of immediate feedback strategy." International Journal of Disabilities Sports and Health Sciences 6.3 (2023): 453-465.

[2] Geary, David C., and Mary K. Hoard. "Learning disabilities in arithmetic and mathematics: Theoretical and empirical perspectives." The handbook of mathematical cognition. Psychology Press, 2005. 253-267.

[3] Powell, Sarah R., Katherine A. Berry, and Sarah A. Benz. "Analyzing the word-problem performance and strategies of students experiencing mathematics difficulty." The Journal of Mathematical Behavior 58 (2020): 100759.

[4] Speed, Laura J., Laura A. Eekhof, and Melina Mak. "The role of visual imagery in story reading: Evidence from aphantasia." Consciousness and Cognition 118 (2024): Article 103645.

[5] Simmons, Fiona Rachel, et al. "Dyslexia and mathematics: Theoretical and empirical perspectives." Journal of Research in Special Educational Needs 8.3 (2008): 153–159.

[6] Hinshaw, Stephen P. ADHD: What Everyone Needs to Know. Oxford University Press, 2018.

[7] DuPaul, George J., and Gary Stoner. ADHD in the schools: Assessment and intervention strategies. Guilford Publications, 2014.

[8] Ganz, Jennifer B. "The use of visual supports to teach children with autism spectrum disorders." Journal of Positive Behavior Interventions 9.3 (2007): 195–207.

[9] Hume, Kara, et al. "Using visual supports to teach social skills to students with autism spectrum disorder: A review of the literature." Research in Autism Spectrum Disorders 6.1 (2012): 425–432.

[10] De Cremer, David, and Garry Kasparov. "AI should augment human intelligence, not replace it." Harvard Business Review 18.1 (2021): 1-8.

[11] Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive NLP tasks." Advances in neural information processing systems 33 (2020): 9459-9474.

[12] Olsson, Fred. "Beyond the Basics: Advanced Retrieval Techniques for RAG Systems: Assessing the impact of sentence-window retrieval and auto-merging retrieval on the performance of a RAG system in a Swedish management consulting company." (2024).

[13] Yu, Qinhan, et al. "MRAMG-Bench: A BeyondText Benchmark for Multimodal Retrieval-Augmented Multimodal Generation." arXiv preprint arXiv:2502.04176 (2025).

[14] Karpukhin, Vladimir, et al. "Dense Passage Retrieval for Open-Domain Question Answering." EMNLP (1). 2020.

[15] Amazon Web Services. "Retrieval Augmented Generation.": Published July 3. https://docs.aws.amazon.com/sagemaker/latest/dg/jumpstart-foundation-models-customize-rag.html

[16] Izacard, Gautier, and Edouard Grave. "Leveraging passage retrieval with generative models for open domain question answering." arXiv preprint arXiv:2007.01282 (2020).