ENGIE4800 - Data Science Capstone

Midterm Report: Spring 2025

Caterina Almazan[1], ChengHsin Chang[2], Claire Yi-Chen Chen[3], Yu-Heng Chi[4] and Param Sejpal[5] ,
Columbia Engineering, NY.

cga2133@columbia.edu[1], cc5211@columbia.edu[2], yc4562@columbia.edu[3], yc4548@columbia.edu[4], pns2129@columbia.edu[5]

Company: L'Oréal

Title: Multi-Modal Search using RAGs

Project Mentor: Professor Sining Chen

Industry Mentor: Rémi Ferreira

Table of contents

1. Introduction 1

2. Literature Review 2

3. Problem Statement & Final Project Goal 3

4. Dataset 4
4.1 EDA

5. Methodology and Structure

5.1 Methods to be tested

6. Acknowledgement

References

1 Introduction

As artificial intelligence transforms e-commerce, L'Oréal aims to enhance product discoverability and user experience through advanced AI-driven search mechanisms. Traditional keyword-based searches often fail to capture complex product attributes, making it challenging for customers to find the right products. Multi-modal search, integrating both text and image inputs, offers a more intuitive and efficient way to navigate vast product catalogs.

Our project leverages **vector-based Retrieval-Augmented Generation (RAG)** to enable intelligent multi-modal search. Using **CLIP (Contrastive Language-Image Pretraining)** for image-text embeddings and **Sentence Transformers** for textual data, we encode product descriptions and images into a unified vector space. The **FAISS (Facebook AI Similarity Search)** index ensures efficient retrieval, allowing users to search by textual queries (e.g., "waterproof black mascara") or images.

For response generation, we integrate the **OpenAI API**, enabling the system to generate detailed, context-aware product descriptions based on retrieved results. This enhances search accuracy and personalization, creating a seamless and intelligent shopping experience for L'Oréal customers.

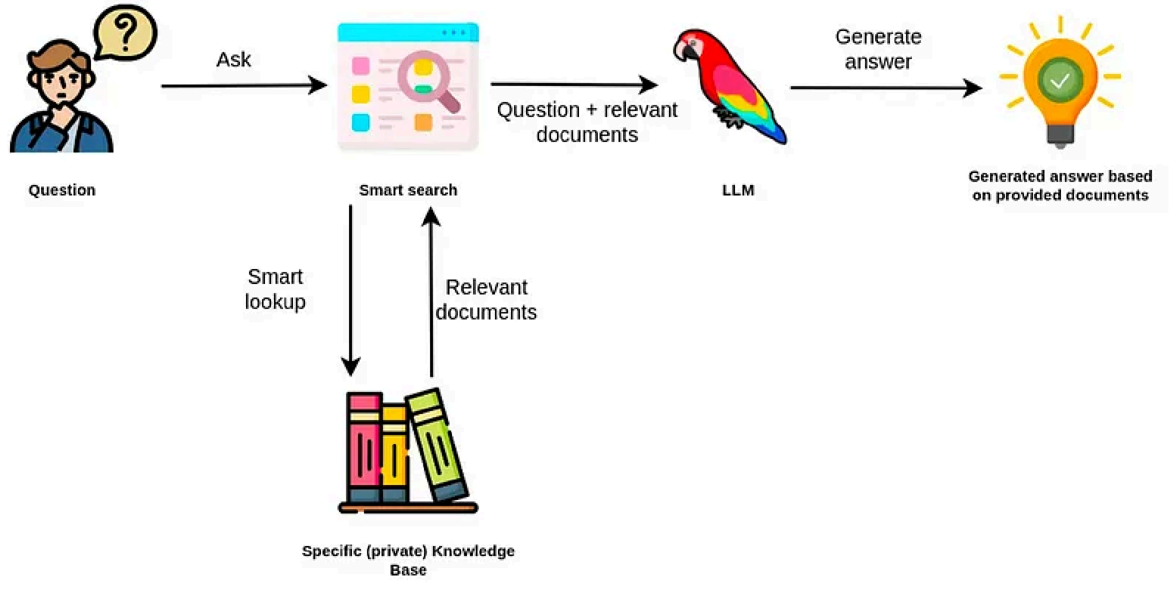

Ask
Question
Smart search
Question + relevant documents
LLM
Generate answer
Generated answer based on provided documents
Smart lookup
Relevant documents
Specific (private) Knowledge Base

Figure 1: Multimodal RAG pipeline

2 Literature Review

Previous data science capstone projects under L'Oréal’s AI initiatives provide critical insights into retrieval-based search methods. These studies compared Vector RAG and GraphRAG in AI-driven product recommendation systems, influencing our approach to multi-modal search.

The Fall 2024 project [1], conducted by Aneri Bijal Modi, Ishita Pundir, Shumail Sajjad, Yash Anish Dange, and Zainab Shakruwala, evaluated Vector RAG vs. GraphRAG for L'Oréal’s chatbot and product recommendation system. Their results showed Vector RAG is faster and better suited for handling broad, unstructured queries, whereas GraphRAG was more effective in understanding hierarchical relationships but was computationally expensive. Given our goal of integrating image-based retrieval, we build on their findings by optimizing Vector RAG’s efficiency in a multi-modal search setting.

Another Fall 2024 project [2], led by Akarsh Rastogi, Leah Uzzan, Nafisa Ali, Rajat Gupta, Tushar Badhwar, and Tushar Prasad, focused on minimizing hallucinations in AI-generated product recommendations by comparing Vanilla RAG and GraphRAG. Their results confirmed GraphRAG enhances precision but requires high processing power, while Vector RAG provides better recall and adaptability, making it preferable for large-scale retrieval. Since our project expands retrieval to both text and image data, we follow their recommendation of prioritizing vector-based retrieval while integrating CLIP embeddings for multi-modal search.

By leveraging these findings, our project extends previous research by integrating multi-modal search with both text and image embeddings, which was not explored in past studies. Unlike prior work that primarily focused on text-based retrieval, our approach incorporates CLIP embeddings for image-driven search, enabling users to find products more intuitively. Additionally, where earlier projects aimed to reduce hallucinations and improve retrieval efficiency, we go further by integrating OpenAI API-driven response generation to enhance contextual understanding and personalization in recommendations. This distinction allows us to refine retrieval mechanisms while ensuring a seamless, dynamic, and intelligent product discovery experience for L'Oréal customers.

3 Problem Statement & Final Project Goal

L'Oréal's portfolio consists of over 36 global brands, creating a vast array of assorted products that all cater to different customer needs and preferences. Thus, it is easy to understand how overwhelming the shopping experience can be as a beauty consumer. With so many options, how can one possibly narrow-down the options in searching for the best product for *you*?

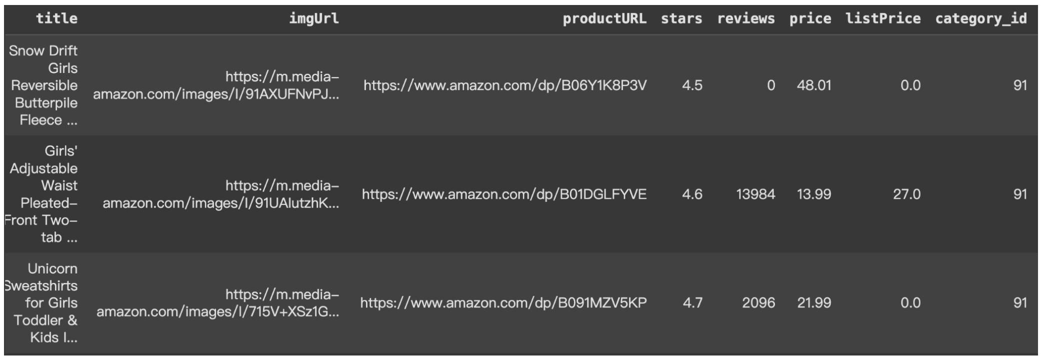
With the average L'Oréal customer in mind, our final goal for this project is to create a functional multi-modal search model that leverages the embedding applications of a vector-based RAG on a diverse product dataset containing both image and text data. Currently, we are applying the vector-based RAG methodology via the embedding underpinnings of the CLIP model. This model will take potential L'Oréal customer queries regarding products as input and outputs certain images of products that best meet the customer's search needs and/or query constraints. For example, if a potential customer where to input a search query such as, "Find me a black waterproof mascara" or "Find a shampoo in a purple bottle that is under 12 oz and less than $20," the goal for our model would be to output the top five (k) images of products which fit these descriptions as closely as possible.

Additionally, we aim to develop a final evaluation metric that will aid in providing classification and differentiate between a good output response and a better model output response. This metric will allow us to more objectively rank the performance of our different models and show which model would provide a better response for L'Oréal consumer interaction prompts.

Furthermore, to promote effective model testing we are in the process of accumulating a comprehensive list of approximately 50 well-defined prompts that would be representative of a L'Oréal customer to ask our multi-modal search model. The goal is that these refined queries will help us create a more accurate, user-friendly product for consumers!

4 Dataset

Our project utilizes a dataset from Hugging Face called "Amazon Product" [3], from which we selected the top 10 product categories. For each category, we extracted 1,000 products, resulting in a total dataset of 10,000 products. And we provided illustrations of the chosen categories, including various clothing, accessories, and toy segments. To enhance our dataset and improve its adaptability to complex queries, we will also explore a previous dataset, Amazon_Reviews_2023_Metadata, which contains detailed product descriptions. By incorporating this metadata, we aim to make our dataset more diverse and robust, enabling more comprehensive analysis and better handling of nuanced queries.



title	imgUrl	productURL	stars	reviews	price	listPrice	category_id
Snow Drift Girls Reversible Butterpile Fleece ...	https://m.media-amazon.com/images/I/91AXUFNvPJ...	https://www.amazon.com/dp/B06Y1K8P3V	4.5	0	48.01	0.0	91
Girls' Adjustable Waist Pleated-Front Two-tab ...	https://m.media-amazon.com/images/I/91UAlutzhK...	https://www.amazon.com/dp/B01DGLFYVE	4.6	13984	13.99	27.0	91
Unicorn Sweatshirts for Girls Toddler & Kids I...	https://m.media-amazon.com/images/I/715V+XSz1G...	https://www.amazon.com/dp/B091MZV5KP	4.7	2096	21.99	0.0	91

Figure 2: Dataset Head

4.1 EDA

To create the word cloud image, we compiled product titles from our dataset into a single text string. We then removed common stopwords and customized the filtering by excluding gender-related words such as "Men," "Women," "Girls," and "Boys" to focus on more descriptive keywords. Using the WordCloud library, we generated a visualization with a "coolwarm" colormap to highlight frequently occurring words. The resulting image provides insights into dominant themes in our dataset, with larger words representing more frequently mentioned product attributes and categories.

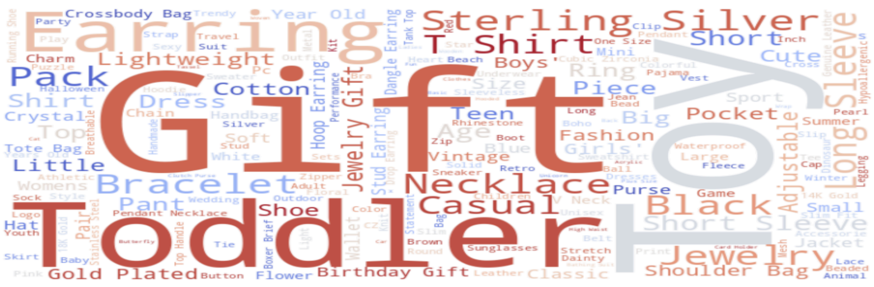


Figure 3: Data Word Cloud

For this visualization, we analyzed the distribution of products across main categories in the Amazon Products 2023 dataset. Using a horizontal bar chart, we visualized category-wise product counts to identify dominant and niche segments. "Clothing, Shoes, and Jewelry" stands out with the highest number of products, followed by "Amazon Home" and "Automotive." This visualization helps highlight category trends and potential areas for further exploration.

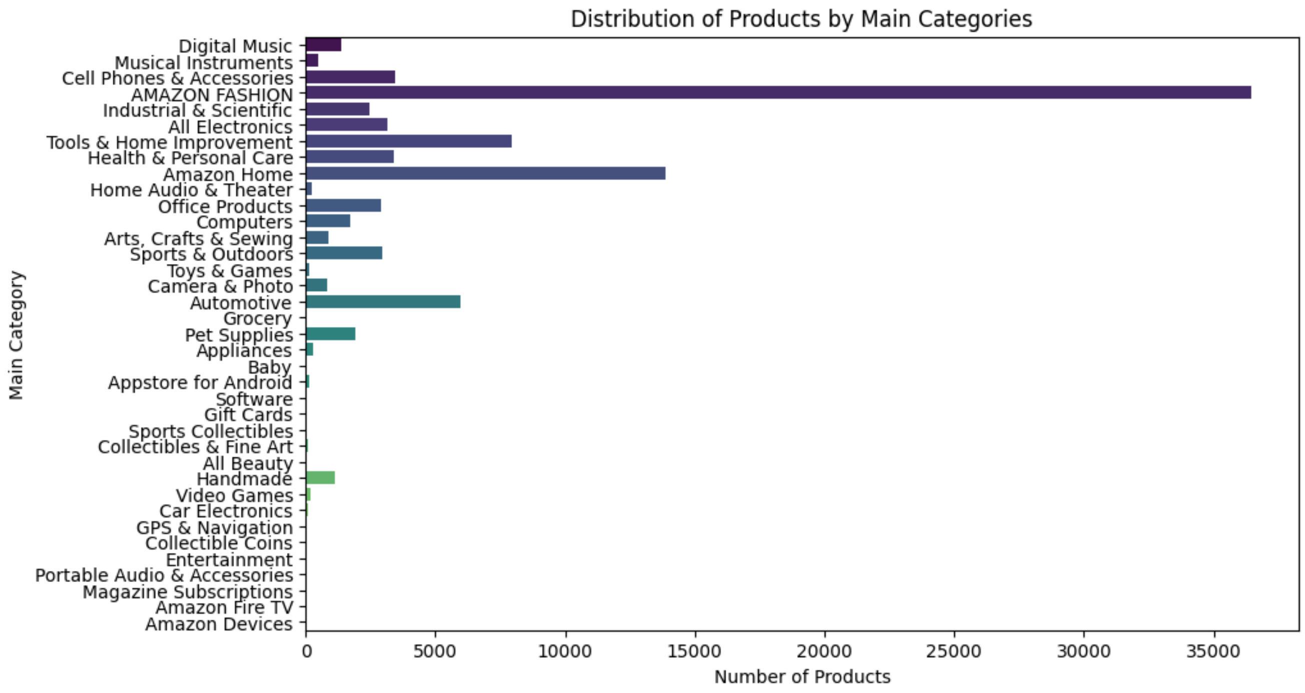
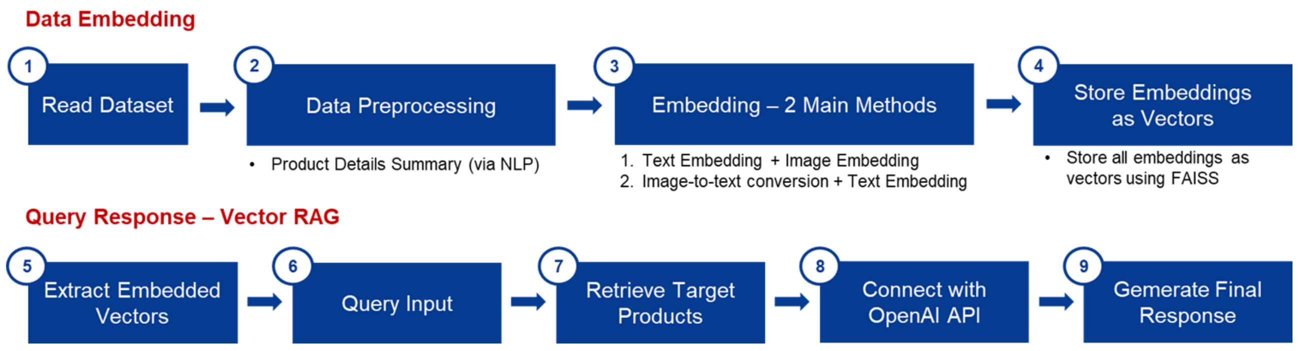


Figure 4: Product Category Distribution

5 Methodology and Structure

As mentioned earlier, our goal is to build a multi-model search model, which acts like a professional sales assistant which can take users' inputs (queries) regarding their desired products, and generate corresponding recommendations (responses). To cater customers' needs of inquiring about L'Oréal products, this model would need to learn information from internal datasets – a L'Oréal product catalog. In our early testing stage, we utilized the Amazon Product Dataset to build and train our model.

To generate human-like responses, we used the OpenAI API to utilize its large language model (LLM), and applied Retrieval Augmented Generation (RAG) to enable the LLM to output results from internal datasets. Generally, Vector RAG and Graph RAG are two main RAG methods adopted by professionals. In this project, Vector RAG was implemented as we aligned with our mentor. Vector RAG is a technique where internal information is first embedded as vectors. Afterwards, when an input comes in, the input will be embedded using the same method, and then compared with the embedded vectors to retrieve required information. Based on the retrieved results and prompts, LLM will generate final responses. Below is the flow chart of the works to give a clearer picture of the overall methodology.


Data Embedding
1 Read Dataset
2 Data Preprocessing
• Product Details Summary (via NLP)
3 Embedding – 2 Main Methods
1. Text Embedding + Image Embedding
2. Image-to-text conversion + Text Embedding
4 Store Embeddings as Vectors
• Store all embeddings as vectors using FAISS
Query Response – Vector RAG
5 Extract Embedded Vectors
6 Query Input
7 Retrieve Target Products
8 Connect with OpenAI API
9 Gemerate Final Response
Figure 5: Project Workflow

There are two main stages in our model. The first stage is data embedding. After the dataset was read, data preprocessing is required due to the lengthy texts in product details. From previous steps in EDA, it is learnt that product details are descriptive, and thus long. If we embed all product details into vectors, it may contain information that is not directly related to the product. For example, there could be a lot of descriptive words or connected words such as "Our product", "And", "Very", "Customer Favorite", etc. These cannot help us to learn the functionality and features of the product, not helping our retrieval results. Therefore, before we embed the data, we plan to utilize Natural Language Processing (NLP) techniques to summarize the product details. The third step in the flow is embedding. Two main methods will be exercised - the first is to do text embedding and image embedding separately and consider both of them in the retrieval function; the second is to convert image into text (using LLM) and then do text embedding purely. There are multiple models available for text and image embeddings. We will test over several suitable models to determine the best combination. We have tried the CLIP model for both text and image embeddings, and found this model performs well on image embedding but not ideal for text embedding. Therefore, Sentence Transformer is the next embedding method we are going to test. More models will be selected and tested gradually as we move on in our project. All embedded vectors will be saved in the vector space using FAISS to prevent long processing time for embedding every time we run the model - this marks the end of the first data embedding stage.

The second stage is the query response from LLM and Vector RAG. Every time we run the model, it first extracts embedded vectors from the stored vector space and waits for users' inputs. When a query comes in, our model will use LLM to summarize the key asks for products from the customer, and embed this summarized text query using the same text embedding technique as the one applied on our dataset. With all information embedded, the model will retrieve the most similar products based on our designed retrieval function, where the comparison is calculated using vector distance. The OpenAI API will take the retrieved similar products and our prompts as input to generate final output as recommendations to the user. We will also test different prompts to ensure final outputs are accurate and could satisfy users' query.

5.1 Methods to be tested

Our next method we intend to test is applying a sentence transformer for embedding on the text data (i.e. product title and product description) and applying a CLIP model on the query and image data; this combination would let us avoid vector embedding misalignment as our query and image embeddings will exist in the same space. Another important model we want to employ is translating the image data to text – essentially describing each product image in a text format which would allow us to apply the same embedding method across all parts of the model. We also plan to experiment with NLP methods and with integrating an OpenAI API to assist with simplifying users' queries in a way our model will better understand the most important portions of the request; however, one main challenge with integrating the OpenAI API is that the number of tokens we would need in order to create an excellent model would be out of the scope of our project budget constraints, whereas CLIP model provides us with essentially unlimited tokens.

Furthermore, given sufficient time we would like to explore other methods for text and image embeddings: ALIGN – a Google-developed CLIP alternative – may prove to be promising as it has been trained on massive-scale noisy web data. Additionally, BLIP-2 (Vision-Language Pretraining) is optimized for long-form descriptions and question-answering which may prove to be beneficial in digesting more complex user queries and product specifications.

5.2 Current Progress - Embeddings and Search Output

Thus far, we have used product titles and product images from the dataset, generating embeddings using the CLIP model. These embeddings have been stored in FAISS files to ensure efficient and sustainable retrieval. A *retrieve_similar_products* function has been designed to take the user query (embedded version) and output number *k* as input, and compare similarity with text and image embeddings to search the most similar *k* products. Some simple queries have been tested, showing strong performance.

In the following query, we tested a simple query to find a silver necklace. With alpha = 0.2, where the retrieval function assigns a weight of 0.2 to text embeddings and 0.8 on image embeddings, the top five retrieved images were perfectly aligned with the query, showing highly accurate results. After multiple trials, we observed that a low but nonzero alpha led to the best outputs in this case.

```
# Sample query
sample_query_3 = "silver necklace"

retrieved_results_3 = retrieve_similar_products(sample_query_3, k=5,
alpha=0.2)
```

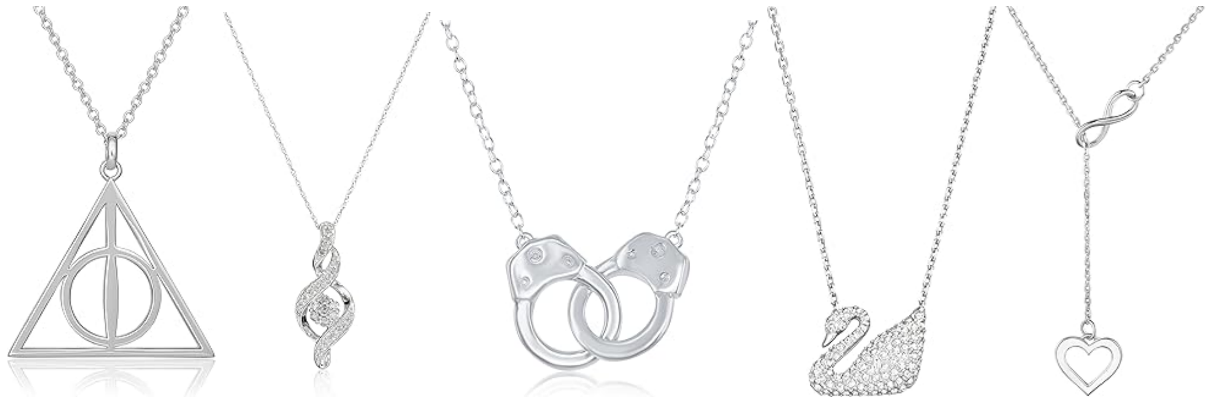


Figure 6: Sample Query Output Images - Set 1

In another test, we used the query "cute girl dinosaur shirt" as input. For this case, the best results were with zero alpha, meaning the output relies entirely on image embeddings. While the retrieved images were generally relevant, they were not as precisely matched as in the previous example, and the last output did not align with the query.

```
# Sample query
sample_query_2 = "cute girl dinosaur shirt"
```

```
retrieved_results_2 = retrieve_similar_products(sample_query_2, k=5,
alpha=0)
```

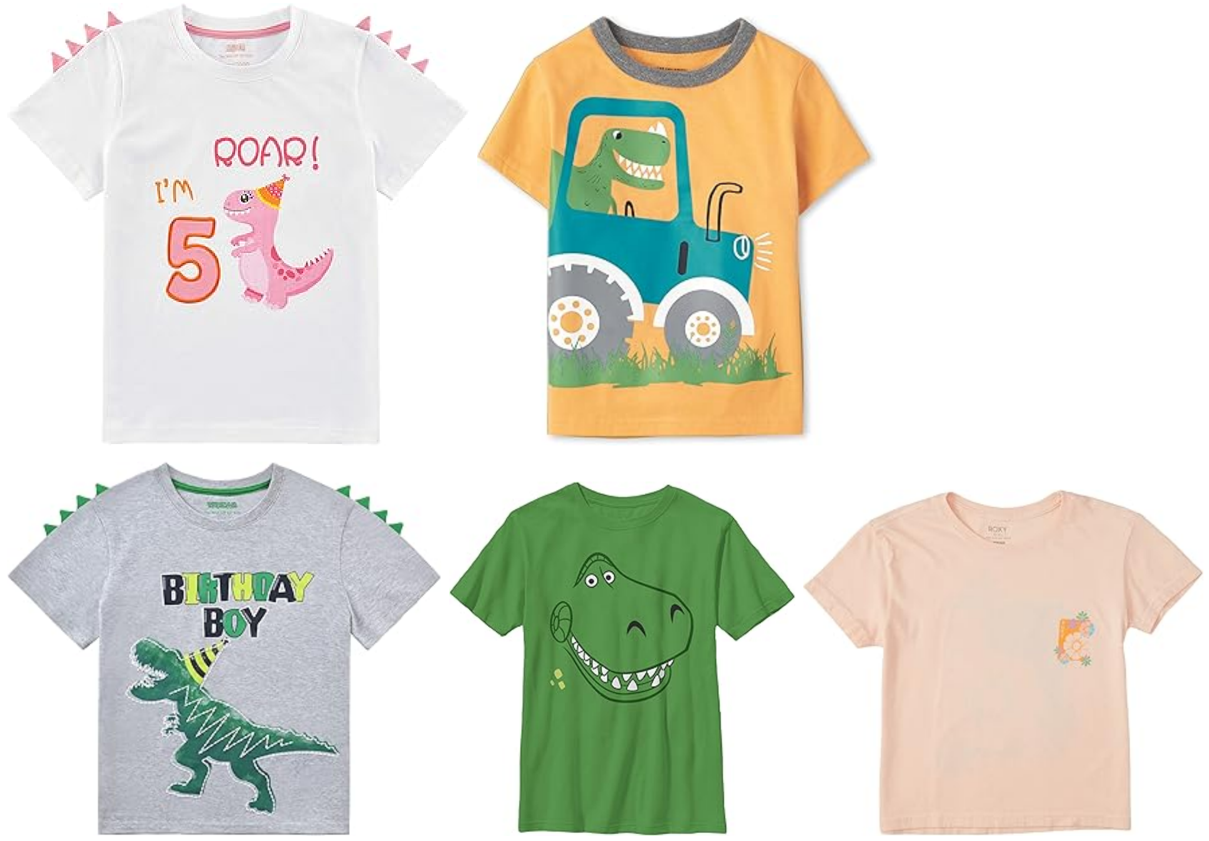


Figure 7: Sample Query Output Images - Set 2

Through these experiments, we observed that the best alpha varies across queries. Additionally, when testing longer queries, we found the retrieval accuracy decreased compared to simple queries. To address this, we plan to explore the optimal alpha using proper evaluation metrics and enhance the performance for longer queries.

6 Acknowledgements

We are sincerely thankful to the L'Oréal Tech Accelerator team, Rémi Ferreira and Nicole Brye, as well as Professor Sining Chen for their mentorship, guidance on project execution and invaluable feedback throughout the course. We also want to thank Savannah Thais for her engaging lectures on AI ethics and the homework assignments that deepened our understanding of responsible data science. Additionally, we appreciate the teaching assistant - Phoebe Chen for her time and support in advising us on course concepts and assisting with project related inquiries.

References:

[1] A. B. Modi, I. Pundir, S. Sajjad, Y. A. Dange, and Z. Shakruwala, "Vector RAG vs. GraphRAG for AI-powered Chatbots and Product Recommendations," Columbia University Capstone Project, Spring 2023. Available: https://github.com/engie4800/dsi-capstone-fall-2024-loreal-ragvsgraphrag

[2] A. Rastogi, L. Uzzan, N. Ali, R. Gupta, T. Badhwar, and T. Prasad, "Minimizing Hallucinations in AI-Generated Product Recommendations: A Comparison of Vanilla RAG and GraphRAG," Columbia University Capstone Project, Fall 2024. Available: https://github.com/engie4800/dsi-capstone-fall-2024-loreal-rag-capstone

[3] Studeni, "AMAZON-Products-2023 Dataset," Hugging Face, 2023. Available: https://huggingface.co/datasets/Studeni/AMAZON-Products-2023

[4] S. Witalec, "Building Multimodal Search and RAG," DeepLearning.AI, 2024. Available: https://www.deeplearning.ai/short-courses/building-multimodal-search-and-rag/