CS6320 Final Project: Gift Findr

James Han[1] and Anna-Duyen Ngo[1]

[1] Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas

Github Repo:

`https://github.com/aan200006/giftfindr`

YouTube Link:

`https://www.youtube.com/watch?v=KwBERkRoszI`

Scope

Create a chat bot that provides gift suggestions by sending links to the user based on their needs.

1 Overall Goal

Our goal is to create useful links and gift recommendations for users trying to find a gift. Given preferences such as age, interests, and budget, the chatbot can provide useful gift suggestions, allowing users to directly click links to bring them to the site to purchase. The user has a convenient way to search for gifts by filtering based on their needs.

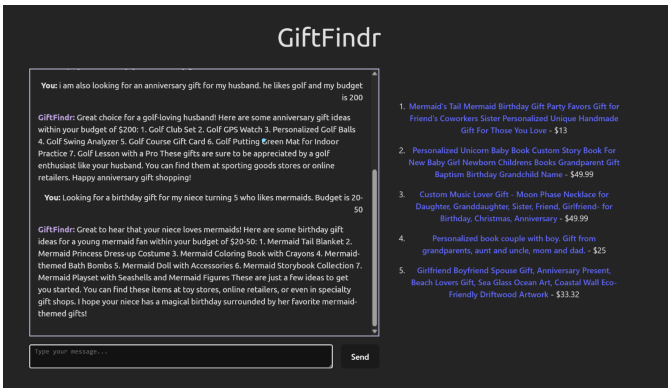

Figure 1. GiftFindr

2 Data Sources

Etsy API

We used Etsy API to obtain active product listings. In our FAISS step, we used web crawling to get 2000 products from Etsy. After running it, the links to the top five recommendations appear and clicking on a link will bring the user to the product on the Etsy website for easy purchase.

OpenAI API

Open AI was used to receive chat responses and structure a JSON object for us. We ask the chat bot to prompt the user for the recipient, recipient's age, recipient's interests, user's budget, and the occasion. So, while the user is speaking to the chat bot, there is a separate call to create a JSON object with those parameters. We use this to restructure a query to give to FAISS.

Facebook AI Similarity Search (FAISS)

First, we crawled 2000 products from Etsy. Then, we created a JSON file with 20 random, unique queries and their relevant gift IDs. We tested a few embedding models to measure how well each embedding captured the semantic meaning of the queries. OpenAI's "text-embedding-3-small" achieved the highest accuracy for relevant gifts as seen in Figure 2. FAISS embeds the products and the query and runs nearest-neighbor algorithm to retrieve the most similar products. We also tested three different FAISS index types: Flat, Hierarchical Navigable Small World (HNSW), and Inverted File Product Quantization (IVFPQ). Flat index is an optimized, exact, brute-force search, HNSW is graph-based, and IVF divides the data into clusters and searches within them. The index is built on the dataset for a quick search. As seen in Figure 3, Flat and HNSW were around the same recall, while IVFPQ did not perform as well. We chose to use Flat index.

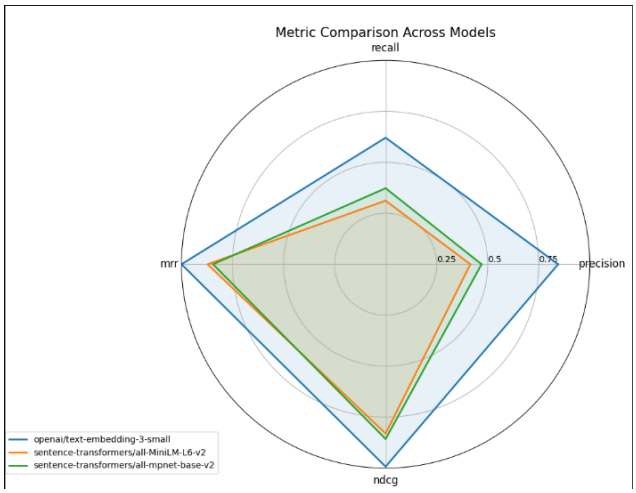

Figure 2. Embedding Model Comparison

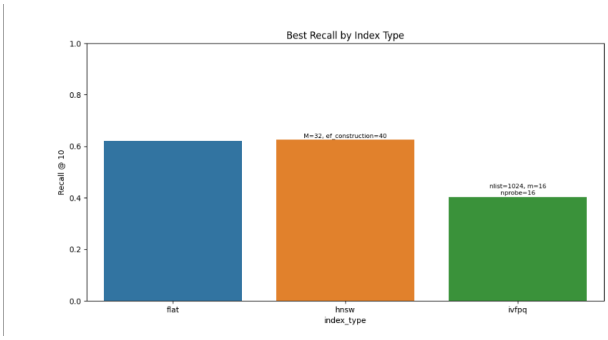

Figure 3. Index Recall Comparison

3 Pipeline

First, the user speaks to the chatbot by giving their preferences. Then, OpenAI will create a JSON object with the relevant information. We restructure the query in the frontend using the JSON to provide a simpler query with an age range and filter by price in the backend. Then, the query is sent to FAISS which returns the top gift recommendations based on the query. This can be seen with a real example in Figure 4.

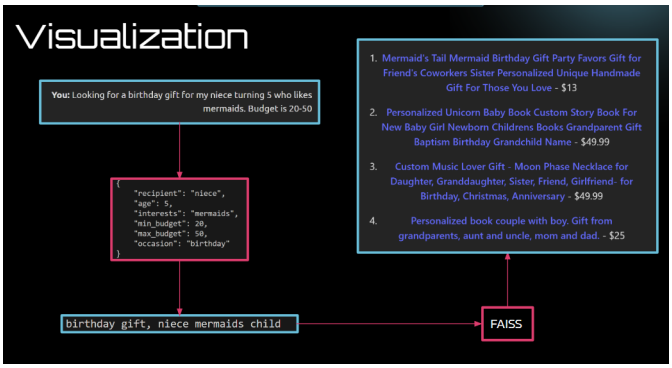

Figure 4. Pipeline Visualization

4 Feedback

Our group of testers first thought the UI was too simple and plain. They suggested adding colors and images to make it more interesting. Another feature they thought to add was a speech-to-text feature and having the chatbot speak out loud to make things simpler. The last critique is to refine the suggestions since they were not always to their liking. The users liked how the website provided the actual links to the products for easy purchase. They stated that is what made it different than simply asking ChatGPT for gift suggestions because ChatGPT does not always provide correct links. One user stated she liked how the chatbot prompted her for specific details. She wouldn't have known what to ask, so it provided guidance on what information to give. Lastly, the users stated they liked the quick response. The chatbot responded quickly, and the users got links very fast. They believed it was much more convenient and faster than if they were to go to the site and manually search for items.

5 Next Steps

UI Update

To make the UI more interesting, we thought of adding animations and images. We attempted to add product images, but did not have enough time to do so. Next, we thought of adding more animations such as a gift box opening with the links or something similar. It is a simple UI at the moment, but adding our additional features will fill it up.

User Accounts

Adding user accounts will allow for saved conversations and gift lists. We thought it would be convenient for the users to save entities, so they can reference back when another occasion occurs. This could be taken a step further by saving occasion dates and sending the user reminders and gift suggestions without requiring the chatbot.

Refine Recommendations

To improve recommendations, we want to implement a likes and dislikes feature. After receiving top recommendations from the query, the user would be able to like or dislike the product so that we can replace the disliked recommendations with a better product. We could either refine the query or exclude products with keywords from the disliked product.

More Products

Due to time constraints, we were only able to include Etsy API, but our reach goal was to include multiple stores like eBay, Amazon, Target, and more. This is where FAISS would be useful. If we got products from all sites and accumulated them, it would be more efficient to retrieve products using FAISS rather than directly calling from each API.

Filtering

To further refine the selection, the user will be able to filter based on other preferences like item availability, estimated shipping date, and shipping prices.

6 Lessons Learned

The most useful lesson learned is to use resources that are available to us. We decided to use FAISS instead of implementing something from scratch because it is already available and efficient. Though we used FAISS, in our own individual testing, we saw that directly getting products from the Etsy API provided better product recommendations. We decided to include FAISS for added complexity, but if we were to deploy the website, directly calling from Etsy or training the model more would provide for better recommendations. In addition, having non-developer testers proved to be useful because they are more focused on the application rather than how it works. There were many feature recommendations and comments about the UI. We also realized having a good design is important. We did not have a designer, but it would be nice to have one provide input on how to make our application better. Although it was working, the main thing users pointed out was the simple design.

7 Contributions

Anna

- **80 points:** Significant exploration beyond baseline.
  - Researched into using Etsy API and OpenAI API to fit the project needs and use in frontend
- **10 points:** Highlighted complexity.
  - Used OpenAI to get information to structure queries and format data to send to FAISS.
  - Restructured query when sending to FAISS
- **10 points:** Discussion of lessons learned and potential improvements.
  - See Sections 6 and 5
- **10 points:** Exceptional visualization/diagrams/repo.

  - Included pipeline and visualized steps. See Section 3
- **10 points:** Discussion of testing outside the team (on 5 people).
  - See Section 4

James

- **80 points:** Significant exploration beyond baseline.
  - Investigating FAISS and running models to achieve best outcome
- **10 points:** Highlighted complexity.
  - Web crawling to gather products from APIs and run them through FAISS.
  - Testing multiple models with FAISS to retrieve the best evaluation
- **10 points:** Discussion of lessons learned and potential improvements.
  - See Sections 6 and 5
- **10 points:** Exceptional visualization/diagrams/repo.
  - Included charts to evaluate performance of models. See Section 2
- **10 points:** Discussion of testing outside the team (on 5 people).
  - See Section 4