ReLM: An Interactive Interface for Web-based Recommendations using Large Language Models

Chris Samarinas
csamarinas@umass.edu
University of Massachusetts Amherst
Amherst, Massachusetts, USA

Hansi Zeng
hzeng@umass.edu
University of Massachusetts Amherst
Amherst, Massachusetts, USA

Zonghai Yao
zonghaiyao@umass.edu
University of Massachusetts Amherst
Amherst, Massachusetts, USA

ABSTRACT

In this paper, we introduce ReLM, an interactive web-based recommendation system that leverages Large Language Models (LLMs) to provide personalized and grounded recommendations for various types of content. ReLM features a user interface for web-based open-domain QA, personalized autocomplete suggestions, and an interactive experience for content discovery. We investigate the personalization capabilities of LLMs through different prompts and natural language narratives to dynamically build user interest profiles. ReLM collects personalized information through conversational interaction with the uers and provides rich multimodal information for the presented results, offering a superior, more interactive, and intuitive user experience for content discovery. We evaluate the effectiveness of ReLM through a user study. Our results demonstrate that personalized suggestions significantly outperform non-personalized suggestions, highlighting the potential of LLMs for recommendations.

1 INTRODUCTION

The ReLM project aims to develop an interactive interface for web-based recommendations using Large Language Models (LLMs). The proposed system provides personalized and grounded recommendations for different types of content. Our core motivation is to explore the capabilities of LLMs to improve content discovery and user engagement. ReLM provides a user interface for grounded recommendations of various types of content, including movies, games, recipes, people, places, and books. It supports web-based open-domain QA with grounded multi-document summarization, display of relevant images, videos, and metadata, and exploration of relevant entities, topics, facets, and questions. The system constructs user interest profiles through interactions with UI and natural language, provides personalized and contextualized autocomplete suggestions, and offers an interactive experience for content discovery using LLMs.

This project will explore the personalization capabilities of LLMs through different prompts across different aspects, such as gender, age, ethnicity, and level of education. The system will also use natural language narratives to dynamically build user interest profiles and allow users to edit them by interacting with the UI or through natural language. Compared to previous recommendation systems, ReLM will collect personalized information from users through conversational interaction with a chatbot and provide rich multimodal information from various sources and APIs for the presented results. Additionally, the system will summarize the user's history to ensure the LLM remains up-to-date and relevant to the user's interest.

The proposed ReLM system will offer a better, more interactive, and intuitive user experience for content discovery. It will augment the current text-based interaction with LLMs with rich structured multimodal information and grounded responses. It will provide superior content recommendations compared to existing traditional Collaborative Filtering systems thanks to LLMs' higher-level understanding of entities, concepts, and their connections. The ReLM system will contribute to the development of a more personalized, interactive and intelligent recommendation system for web-based content.

The implementation for the ReLM project includes designing and developing a user interface for web-based recommendations using LLMs (such as GPT3 and ChatGPT). The system will be based on various prompt templates and a database for storing user profiles and interactions. User feedback will be collected through like/dislike buttons in the interface and/or natural language, and ReLM will update the user's profile and refine the recommendations over time. Our system will be evaluated through user testing and human evaluation. An example use case can be seen in figure 1.

Given that our goals like "personalized", "interactive" and "intelligent" are the subjective requirements where it is difficult to quantify without human judgment, our evaluation will focus in two directions, human evaluation and LLM evaluation. We will follow the traditional human evaluation steps and let some different target users rate their interaction experience with the chatbot in multiple aspects. And we are especially interested in some exploration and discussion about Human-AI-in-the-loop to improve the human evaluation process. Some topics we can explore like 1) Can ChatGPT imitate different target users well? Will human evaluation be in line with LLM evaluation results if we ask ChatGPT to imitate that target users? 2) What if we use HCI methods like Personas, Mental Models and Conceptual Models for target users imitated by ChatGPT?

The reason we chose this specific project is because all of our team members us work in Natural Language Processing and Information Retrieval.

2 RELATED WORK

2.1 Recommendation Systems

Recommendation systems are intelligent algorithms that aim to recommend items to users based on their preferences and past behaviors [14]. These systems have become increasingly important in recent years due to the vast amount of information available on the internet, making it difficult for users to find the information they need. In this article, we will provide an overview of recommendation systems and discuss some of the popular approaches used in developing these systems.

Recommendation systems can be broadly classified into three categories: content-based, collaborative filtering, and hybrid systems [17]. Content-based systems recommend items to users based on their past preferences or behavior, while collaborative filtering systems recommend items based on the preferences of similar users. Hybrid systems combine both content-based and collaborative filtering approaches to provide personalized recommendations to users.

Content-based recommendation systems are based on the idea of recommending items that are similar to items the user has previously liked. These systems use features such as the genre, director, or actors of a movie to recommend similar movies to the user. The primary advantage of content-based systems is that they can recommend items that are unique to the user's tastes, even if those items are not popular or well-known [12].

Collaborative filtering recommendation systems, on the other hand, recommend items based on the preferences of similar users. These systems use historical user-item interactions, such as ratings or reviews, to find patterns and similarities among users. Collaborative filtering can be further divided into two sub-categories: user-based and item-based collaborative filtering. User-based collaborative filtering recommends items to a user based on the preferences of similar users, while item-based collaborative filtering recommends items to a user based on the similarities between items [16].

Hybrid recommendation systems combine both content-based and collaborative filtering approaches to provide personalized recommendations to users. These systems have been shown to provide better recommendations than either content-based or collaborative filtering alone. Hybrid systems can also overcome some of the limitations of content-based and collaborative filtering systems, such as the cold start problem [3].

There are several approaches to developing recommendation systems, including matrix factorization, deep learning, and graph-based approaches [20]. Matrix factorization is a popular approach used in collaborative filtering systems. It involves decomposing a user-item interaction matrix into two lower-dimensional matrices, one for users and one for items. Deep learning approaches, such as neural networks, have also been used to develop recommendation systems. These systems can capture complex patterns in user-item interactions and provide more accurate recommendations. Graph-based approaches, such as graph convolutional networks, use the relationships between users and items to make recommendations. These systems have shown promising results in cold start scenarios, where there is little to no user-item interaction data available [19].

In this work we will explore a new direction in recommendations that relies on Large Language Models and provides a natural language interface for content discovery that makes users' interactions with the system more intuitive and transparent.

2.2 Large Language Models

Large language models (LLMs) are AI systems that have gained attention for their ability to generate high-quality text, perform language tasks, and simulate human-like conversation [6]. LLMs, like OpenAI's GPT-3 and Google's BERT, are built using neural networks and have demonstrated impressive performance on various language tasks [2, 7, 10].

However, LLMs have faced criticism for perpetuating biases and potential misuse in generating fake news [8, 13]. Researchers have proposed techniques to mitigate biases and improve transparency, such as using diverse training data and explainable AI (XAI) [4, 18].

In this work, we develop a recommendation system using ChatGPT, an instruction fine-tuned variant of GPT-3, through carefully designed prompts to support various operations.

2.3 Narrative-driven Recommendations

Narrative-driven recommendation systems use storytelling to enhance user experience, providing context and relevance to recommendations. They differ from traditional systems that rely on user data, item data, or collaborative filtering [15].

Traditional systems often suffer from cold-start problems and ignore the context behind user preferences [5]. Narrative-driven systems address these limitations by using storytelling techniques, personal anecdotes, or fictional narratives relevant to user preferences [1].

Examples of narrative-driven systems include storytelling-based movie and music recommender systems [9, 11]. These systems offer advantages such as a more engaging experience, addressing cold-start problems, and improving user trust in recommendations [19, 20].

In conclusion, narrative-driven recommendation systems use storytelling techniques to enhance user experience, providing context, addressing cold-start problems, and improving user trust. They have the potential to revolutionize recommendations for products, services, and content.

3 PROBLEM STATEMENT

3.1 Problem Statement

In the current landscape of web-based recommendations, there is a need to improve personalization, content discovery, and user engagement. Existing systems largely rely on text-based interactions and traditional Collaborative Filtering methods, which have limitations when it comes to understanding users' interests, preferences, and the connections between different entities and concepts. As a result, these systems may fail to deliver a truly personalized and interactive experience, leaving users feeling unsatisfied with the recommended content. To address this issue and harness the potential of Large Language Models (LLMs), the ReLM project seeks to design and develop an interactive web-based recommendation system utilizing LLMs for a more personalized and grounded content-discovery experience through conversational natural language.

3.2 Proposed Solution

ReLM aims to provide a user interface that enables personalized and grounded recommendations for various types of content through conversational queries. By leveraging the capabilities of LLMs, such as GPT-3 and ChatGPT, our system offers a more interactive and intuitive user experience. We designed an interactive user interface and a database for storing user profiles and interactions. ReLM

uses various prompt templates to create personalized content suggestions, which are refined over time based on user feedback (e.g., like/dislike buttons and natural language input). Our goal is to contribute toward the development of a more intuitive, personalized, interactive, and intelligent recommendation system that addresses many limitations of existing systems.

4 SYSTEM DESCRIPTION

4.1 Usage Flow

In our final high fidelity prototype, the user's flow consists of four main stages, as outlined below.

Stage 1: The primary interface takes inspiration from modern search engines with an emphasis on simplicity. We provide a text hint in the search bar, indicating the types of queries ReLM supports to differentiate it from traditional search engines. The user's interaction with our system can be seen in Figure 1.

Stage 2: Users type queries in the search box, receiving contextual query completions while typing as seen in stage 2 of Figure 1. The autocomplete suggestions are generated based on the user's interest profile and search history.

Stage 3: Pressing enter triggers the search, revealing a result panel as shown in stage 3. These results are supplemented with images. When a user clicks on a result, additional details about the entity are displayed (as seen in stage 4). Like and dislike buttons are included for our explicit feedback mechanism to refine the recommendations. Clicking 'more suggestions' expands the results panel, revealing additional recommendations based on the feedback provided. We also present automatically generated facets that users can click to explore other aspects of their query. For instance, if the user clicks on 'food', a new search query 'thing to do in spain / food' will be executed.

Stage 4: Clicking an entity result reveals additional details about it in stage 4. A larger image along with a more detailed description is displayed. At the bottom, similar recommendations to the displayed entity are presented, and the user can refresh the suggestions with the refresh icon. The user can easily return to the result panel by clicking the close button on the top right.

Search History Located on the top right of the page is a user profile icon. When clicked, it displays recent search history in the first tab, as shown in Figure 2. Users can close this panel either by selecting a query or clicking the close button. In the second tab next to history, there is a user-editable profile, where users can explicitly describe their profile and preferences to fine-tune the suggestions of ReLM (as seen in figure 3).

4.2 Features

In this section, we discuss various features of our system, which include query-based item recommendations, item-based recommendations, entity summaries, query intent classification, autocomplete, query facet suggestions, and user profiling personalization. These features are designed to provide users with a personalized and efficient search experience. We utilize ChatGPT to generate recommendations, summaries, and personalized suggestions based on user profiles and interaction data. Additionally, we employ the Bing Entity Search API and Bing Image Search API to ground the recommendations and retrieve relevant metadata. By combining these techniques, our system offers a dynamic and adaptive search experience that caters to users' unique preferences and behaviors.

4.2.1 Query-based item recommendations. The query-based item recommendations feature suggests items to users based on their query. It uses the keywords or phrases provided by the user to generate a list of relevant items that match the query. The recommendations can be based on various factors such as popularity, relevance, and personalization, depending on the system's design and implementation. In our system, we utilized ChatGPT to provide entity suggestions for any given query with the following prompt:

```
User summary: {}

Based on the user's summary, give 5
    suggestions for the query '{}'. Return a
     list of json objects in this format:

[{{"title": ..., "description": ...}}, ...]

the titles should only be entity names, and
    only return the json, no other words in
    your response
```

The prompt template has 2 variables for the user's profile summary and the target entity. The user's summary helps adapt the suggestions of the LLM so that they match both with the target entity and the user's general interests. As we will see later in section 2.7, the user summary is generated by ChatGPT using all their interactions and their optional defined profile (bio).

To ground the recommendations generated from ChatGPT, we use the Bing Entity Search API for every entity name. For location entities, we save the entity name, search result URL, entity home page URL, entity address, and entity telephone. For other entities, we save the entity name, search result URL, entity home page URL and entity description. We then use the Bing Image Search API to retrieve images for all the grounded entities. Even though in our current prototype we don't display any special metadata for the entities, in a future version we plan to integrate them.

4.2.2 Item-based recommendations. The item-based recommendations feature suggests items to the users based on a target item/entity and their profile. The system identifies items that are similar to the target and the ones that the user has interacted with, and recommends them as potential options. Item-based recommendations can be implemented using various techniques such as collaborative filtering, clustering, and similarity-based algorithms. In our system, we also employed ChatGPT for item-based recommendations using the following prompt:

```
User summary: {}

Based on the user's summary, give 4 other
    suggestions of the same entity type for
    someone who likes '{}'. Use the
    following json format:

[{{"title": ..., "description": ...}}, ...]
```

```
the titles should only be entity names, and
    only return the json, no other words in
    your response
```

Similar to the query-based item recommendations, we then ground all the suggestions using the Bing Entity Search API and retrieve additional metadata depending on the type of the entity, including a representative image.

4.2.3 Entity summaries. Entity summaries are concise descriptions of specific entities such as people, places. The goal of entity summarization is to provide a brief and informative summary of the entity's most important attributes or characteristics. This is often done using techniques such as entity extraction and entity linking, which identify and link the entity mentions in the text to their respective entities in a knowledge base or database.

We first used the Bing Web Search API retrieve the top 10 documents for the given entity query and combine them as input context for ChatGPT. We then used the following prompt to get a personalized concise entity summary:

```
User summary: {}
Web context: {}

Based on the user's summary and the web
    context, write a personalized summary
    about '{}' in around 40 words, Use the
    following json format:

{{"title": ..., "description": ...}}

only return the json, no other words in your
     response
```

4.2.4 Query Intent Classification. We have 2 main categories of queries; recommendation queries and entity-oriented queries. The recommendation queries expect a list of suggested items with brief description as output, while the entity-oriented expect a more detailed summary about some specific entity. In order to identify the query intent we used the following prompt:

```
Choose the most appropriate intent below for
     the query '{}':

1. user wants some recommendations
2. user wants information about something

Output format should be only the number
```

4.2.5 Autocomplete. Autocomplete is aimed at enhancing user experience by providing real-time query suggestions. This feature is also powered by ChatGPT. ChatGPT is configured with a tailored prompt comprising the current query prefix, user's history of queries and the instruction. For example, if the query prefix is "best restaurants in", ChatGPT generates possible completions such as "best restaurants in New York" or "best restaurants in San Francisco" based on the user's query history.

This feature provides users with an intelligent and personalized search experience, dynamically adapting to their search patterns and preferences to offer the most relevant auto-completed queries. The following prompt is used to generate auto-complete suggestions:

```
Generate 5 `completed queries` based on the
    given `query prefix` and taking into
    account the user's `search_history`.
    Ensure that the `completed queries`
    cater to the user's personalized needs
    as encoded in their search_history.
    Present the `completed queries` as a
    list of strings in Python list format
    without including any additional
    information. Each query should be
    enclosed in double quotes and separated
    by commas.
search_history: {}
query prefix: {}
completed queries (example format): ["item
    1", "item 2", ...]
completed queries:
```

4.2.6 Query Facet Suggestions. The 'query facet suggestions' feature is an integral part of the ReLM system, aiming to provide users with a more personalized and targeted content discovery experience by offering relevant aspects of their queries. This feature utilizes ChatGPT with a prompt template containing two variables: the user's summary and the input query. By incorporating the user's preferences and interests, along with the context of the search query, the system dynamically generates personalized facets for the user to explore. More specifically we used the following prompt template:

```
Given the user's summary and query, generate
     up to 5 diverse and distinct topics/
    facets, the topics should be short (up
    to 3 words) using the json format below:

User summary: {}
Query: {}

{{"facets": ["facet 1", "facet 2", ...]}}

only return the json, no other words in your
     response
```

For example, if a user searches for "best things to do in Spain," the system might present facets such as "landmarks," "cuisine," or "festivals," taking into account the user's specific interests, like architecture, food, or cultural events. This feature not only allows users to narrow down their search but also enhances the overall content discovery journey through tailored and meaningful suggestions.

4.2.7 User profiling & Personalization. User profiling & Personalization is crucial to deliver a search experience tailored to each user's unique preferences and behaviors. The goal of this feature is to create a comprehensive user profile, leveraging all the data accumulated during their interactions with our system. The interaction data comprises clicked items, liked items, disliked items, and past queries.

To implement this, we utilize ChatGPT to generate a user summary based on all the interaction data. This provides a concise and coherent overview of the user's activity and preferences, enabling our system to deliver more personalized and relevant search results.

Anytime a user has a new interaction with the system, the user profile is updated. This means our system remains responsive and adaptive to changing user behaviors and preferences, ensuring an always up-to-date personalized search experience.

This is the prompt for generating user summaries:

```
Generate a concise summary that profiles the
     user based on their `bio` and `
    history_queries`, `clicked_items`, `
    liked_items` and `disliked_items`.
    Ensure that this summary accurately
    captures the user's preferences and
    interests.
bio: {}
history_queries: {}
clicked_items: {}
liked_items: {}
disliked_items: {}
summary:
```

Filling out all placeholders using the user's interaction data enables us to prompt ChatGPT to produce a concise and coherent summary in natural language. We present below an example of a user summary generated by ChatGPT (due to the page limit, we omit the user interaction data):

> "John Doe, a 35-year-old residing in San Francisco, shows a deep enthusiasm for technology, travel, fitness, and cooking. He actively seeks updates on the latest in technology and is intrigued by travel destinations that have a tech angle. His fondness for cooking is mirrored in his inclination towards healthy recipes. While his interest in fitness is apparent, a specific video tutorial on proper squat technique failed to captivate him. His preferred content spans tech podcasts, travel blogs with a focus on tech-centric locales, and recipes promoting a balanced diet."

5 EVALUATION

To evaluate the effectiveness of our language model-based recommendation system, we conducted a user study comparing suggestions with and without personalization. We recruited 20 participants, consisting of friends and family members, aged between 18 and 65 years old. Each participant was asked to write a short bio about themselves, such as:

> "I am a PhD student from UMass Amherst, my major is computer science. I am from China, and I like things with a sense of technology. I love all kinds of sports. I also like natural scenery, like traveling by car, and often go hiking."

Using these bios, we generated personalized recommendations for 10 different queries. Participants were then asked to rate each suggestion on a scale of 1 to 5, with 1 being the lowest and 5 being the highest. To compare the effectiveness of personalized and non-personalized suggestions, we performed a one-sided t-test on the average ratings.

6 RESULTS

The results of the t-test indicated that there is a significant large difference between personalized and non-personalized suggestions. The mean rating for personalized suggestions was 4.52, while the mean rating for non-personalized suggestions was significantly lower at 3.5 (as seen in table 1). The t-test yielded a t-value of 15.2 with 19 degrees of freedom, and a p-value less than .001.

Setting	Average	St. Deviation
Not Personalized	3.5	0.3162
Personalized	4.52	0.3033

Table 1: Comparison of average ratings and standard deviations for personalized and non-personalized suggestions

These results demonstrate that the language model-based recommendation system with personalization significantly outperforms the non-personalized version. Participants found the personalized suggestions to be more relevant and useful, as evidenced by the higher average ratings. One good example of personalized recommendations can be seen in table 2. This suggests that incorporating personalization into the recommendation system can lead to a more satisfying user experience.

7 CONCLUSIONS

In this paper, we presented ReLM, a language model-based content recommendation system that leverages the power of Large Language Models (LLMs) to provide personalized and grounded recommendations for various types of content. Our results demonstrated that personalized suggestions significantly outperformed non-personalized suggestions, with a mean rating of 4.52 compared to 3.5 (as seen in table 1). This indicates that incorporating personalization into the recommendation system leads to a more satisfying user experience and better overall performance.

As LLMs continue to become the core of search and recommendation systems, user profiles will likely evolve into summarized prompts for these models. ReLM serves as an early demonstration of the next generation of recommendation systems, showcasing the potential of personalized proactive assistants as an interesting direction for future research and development.

Example - Not Personalized	Same Example - Personalized
Query: best movie to watch **User Profile:** I am a phd student from umass amherst, my major is computer science. I am from China, and I like things with a sense of technology. I love all kinds of sports. I also like natural scenery, like traveling by car, and often go hiking.	**Query:** best movie to watch **User Profile:** I am a phd student from umass amherst, my major is computer science. I am from China, and I like things with a sense of technology. I love all kinds of sports. I also like natural scenery, like traveling by car, and often go hiking.
"The Shawshank Redemption": Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.	"The Social Network (2010)": A gripping tale that portrays the creation of Facebook by Harvard undergrad Mark Zuckerberg. It dives into the drama, politics, and ethical dilemmas of tech startups, making it an exciting and enlightening watch for a computer science student.
"Inception": A skilled extractor is offered a chance to regain his old life as payment for a task considered to be impossible.	"Ex Machina (2014)": A deeply thought-provoking film that explores artificial intelligence and the Turing Test. Its techno-thriller vibe and philosophical undertones about humanity and consciousness will appeal to your interest in technology.
"Parasite": Greed and class discrimination threaten the newly formed symbiotic relationship between the wealthy Park family and the destitute Kim clan.	"Interstellar (2014)": Directed by Christopher Nolan, this film combines science fiction with stunning visual effects of space and otherworldly landscapes, offering a thrilling journey that will satisfy your love for natural scenery.

Table 2: Comparison of non-personalized and personalized movie suggestions

8 FUTURE WORK

There are several areas for improvement and further exploration in the ReLM system:

- Display more entity metadata to provide users with richer information about the recommended content.
- Implement a user authentication system to enable personalized recommendations across multiple devices and sessions.
- Tune and self-host a smaller LLM (e.g., Vicuna-13b) for faster inference and improved system performance.
- Implement suggestion streaming to decrease latency and provide a more seamless user experience.
- Develop near lossless compression techniques for user profiles to optimize storage and retrieval efficiency.

9 PARTICIPATION

Chris contributed to the project by working on the frontend and UI development using React. He also assisted with the backend system design and wrote and reviewed several parts of the report, including the evaluation strategy. Additionally, he contributed to the query facet generation in the backend.

Hansi focused on backend development, taking responsibility for implementing the autocomplete and personalization features using ChatGPT. He also authored the relevant sections about these components in the report.

Zonghai worked on the backend as well, handling query intent classification, item recommendations using ChatGPT, and grounding using the Bing Entity Search API. He documented these aspects in the appropriate parts of the report.

REFERENCES

[1] Hossein Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Narrative-driven recommendations: A taxonomy and survey. *User Modeling and User-Adapted Interaction* 27, 5 (2017), 393–444.

[2] Daniel Adiwardana, Minh-Thang Lu, Demis Hassabis, Antoine Bordes, Volodymyr Mnih, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977* (2020).

[3] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering* 17, 6 (2005), 734–749.

[4] Emily M Bender and Timnit Gebru. 2021. On the dangers of stochastic parrots: Can language models be too big? *arXiv preprint arXiv:2104.04430* (2021).

[5] Jes'us Bobadilla, Fernando Ortega, Antonio Hernando, and Alberto Guti'errez. 2013. Recommender systems survey. *Knowledge-based systems* 46 (2013), 109–132.

[6] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[9] Seonwoo Kwon, Youngmee Kim, and Kyogu Lee. 2018. Storytelling-based movie recommender system. *Multimedia Tools and Applications* 77, 1 (2018), 849–868.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[11] Rui Liu and Lin Chen. 2019. A story-based music recommendation system with deep learning. *IEEE Access* 7 (2019), 89156–89167.

[12] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[13] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[14] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Recommender systems handbook*. Springer, 1–35.

[15] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. *Introduction to recommender systems handbook*. Springer US.

[16] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[17] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[18] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2019).

[19] Xiaoyu Wang, Shuai Wang, Xinyu Zhou, and Quanquan Gu. 2019. Knowledge graph and deep learning aided recommendation: A survey and future challenges. *IEEE Transactions on Knowledge and Data Engineering* 32, 8 (2019), 1534–1558.

[20] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

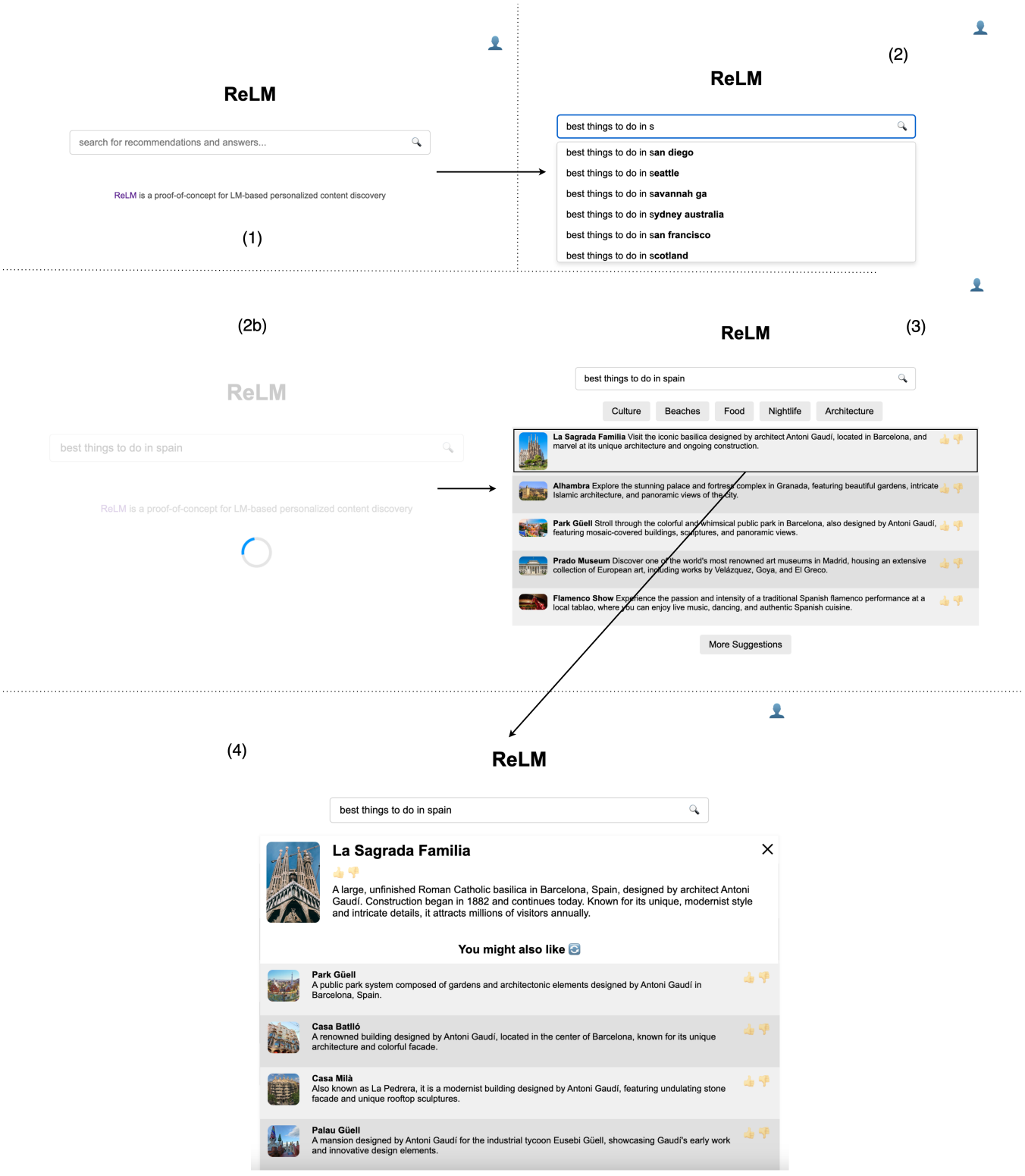

Figure 1: The flow of a user engaging with ReLM encompasses 4 primary stages: 1) inputting queries in a search bar, 2) receiving contextual autocomplete suggestions driven by user interest and search history during typing, 3) obtaining a results panel containing enhanced content, feedback choices, and facets for query exploration, and 4) accessing detailed entity summaries overlaying the results panel when selecting a result, complete with pertinent metadata and related recommendations, allowing effortless navigation using refresh and close icons.

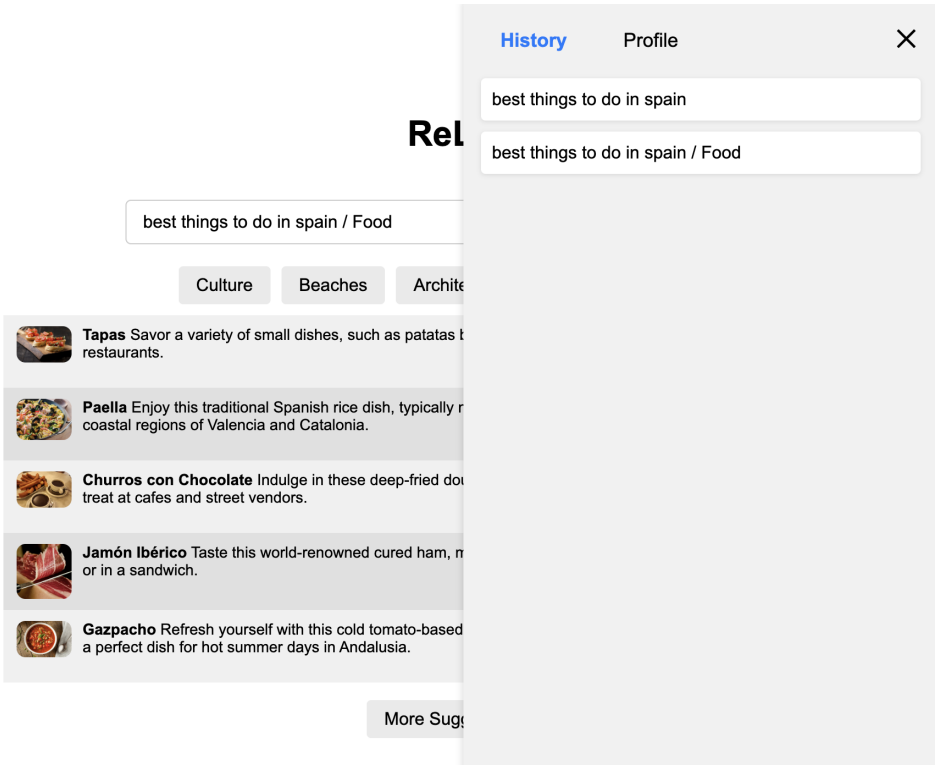

Figure 2: Showing user search history in a sidebar after clicking the profile icon on the top right of the page.

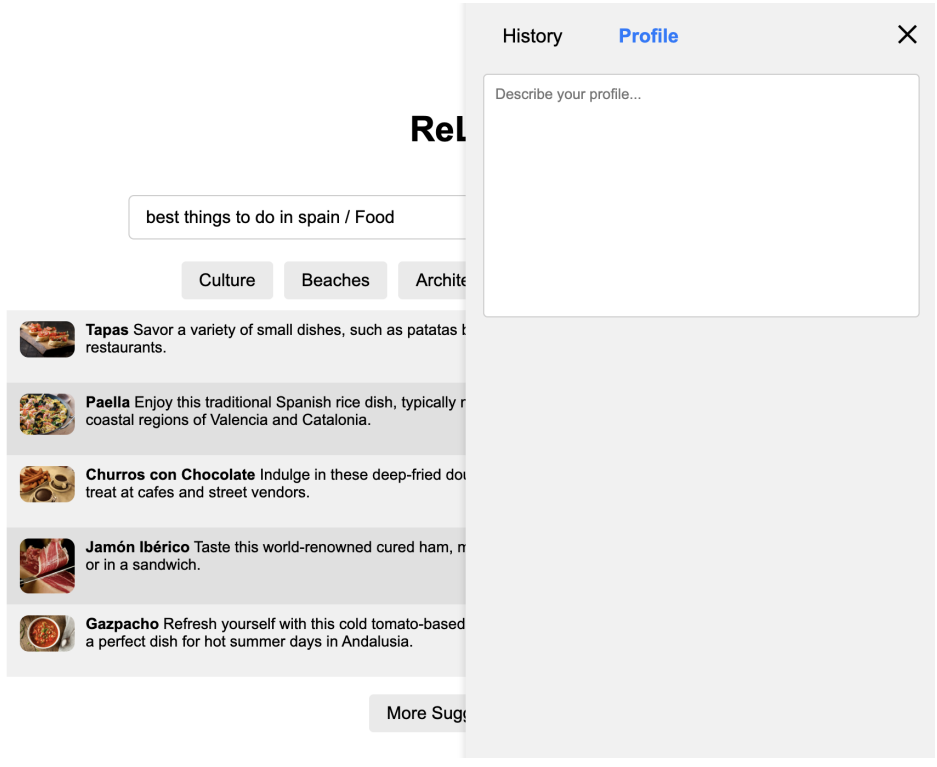

Figure 3: The editable user profile text area next to history.