



POLITECNICO DI TORINO

Master degree course in Computer Engineering - Artificial
Intelligence and Data Analytics

Master Degree Thesis

From Analysis to Application: Employing AI to Enhance User Experience at ESA

Supervisor

prof. Eliana Pastor

Candidate

Samuele LO TRUGLIO

matricola: 295285

Internship Tutor

ESA-ESRIN

dott. ing. Mattia Stipa

ANNO ACCADEMICO 2022-2023

This work is subject to the Creative Commons Licence

To my grandparents

Summary

In the current era, Artificial Intelligence (AI) has gained remarkable traction, revolutionizing tasks such as translation, summarization, and text generation. This master’s thesis delves into the application of AI technologies to enhance User Experience (UX) in European Space Agency (ESA) applications, specifically those managed by EOP-GES (Ground Segment). The thesis comprises two projects and a demonstrator designed to showcase the practical implications of these projects in real-world scenarios. The first project, EO Recommender, uses Earth Online visitors’ navigation history to predict the most likely pages they will visit. The activity includes collecting and curating an ad-hoc EO Clicks dataset capturing user activity on the website. We then employ state-of-the-art recommendation systems such as Gru4Rec and SLiRec. We provide extensive benchmarking and hyperparameter tuning and assessment. The EO Recommender’s development yields valuable insights for ESA and demonstrates the potential for enhancing user engagement through this technology. The second project, the EOP-GES Assistant, introduces an offline virtual assistant powered by Large Language Models (LLMs), which draw knowledge from private document collections, ensuring security. We adopt an open experimentation approach to enable the testing of such a system with real data. Specifically, we collect prompts and corresponding related information and documents from ESA staff. We then enhance and generate new prompts to enlarge the experimental dataset. Finally, we benchmark multiple LLMs and analyze their results in terms of quality, time, and robustness. Models benchmarking provide insights into resource requirements and trade-offs for implementing similar systems. Among the tested LLMs, LLaMA V2 models reveal their remarkable capacity to extract nuanced information from documents of diverse genres. Our analysis and insights offer a promising starting point for further research and implementation endeavors at EOP-GES.

Acknowledgements

I would like to express my deepest gratitude to Mattia Stipa, my ESA tutor, for believing in me even before our first meeting and giving me the opportunity to do this experience.

I'm also grateful to Eliana Pastor, my supervisor, who introduced me in the other side of University life and trusted me for this thesis.

Thanks to the whole EOP-G for including me despite the differences and the short time I spent at ESA and provided me many good inputs for the future.

I'd like to acknowledge the interns and my university colleagues for sharing many coffee breaks and adding a little levity to these experiences.

I am grateful beyond words for the unwavering love, support, and presence of my girlfriend, throughout this master's thesis. Her encouragement has been a guiding light, helping me overcome challenges, distance, and uncertainty.

I could not have undertaken this journey without my friends, all of them. From my old-standing classmates to the newest additions to my life, their presence has been instrumental in shaping my experiences and memories.

Last but not least, this endeavor would not have been possible without my family for their belief in me, that has kept my spirit and motivation high during this process. Thanks should also go to my nephews and my niece for always making me feel special. I would also like to thank my dog Zoe for all the time we spent together and the emotional support.

Contents

List of Tables	10
List of Figures	11
1 Introduction	13
1.1 Problem Overview	13
1.2 Research Goal	14
1.3 Proposal Overview	14
2 Related Work	17
2.1 Background	17
2.1.1 Recommender Systems	17
2.1.2 Virtual Assistants	20
2.2 Existing solutions	21
2.2.1 Earth Online	21
2.2.2 Virtual Assistants	25
2.2.3 Limitations and Drawbacks	26
2.2.4 Justification for New Approaches	28
3 EO Recommender	31
3.1 Dataset	31
3.1.1 Data Source	31
3.1.2 Data Cleaning	33
3.1.3 Data Preparation	34
3.1.4 Dataset Creation	37
3.2 Recommendation System	39
3.2.1 Exploring Recommenders	40
3.2.2 Model Selection	40
3.2.3 Hyper-Parameter Tuning	42

3.2.4	Results	43
4	EOP-GES Assistant	47
4.1	Exploring PrivateGPT	47
4.2	Experimentation	49
4.2.1	Methodology	50
4.2.2	Evaluation	52
4.2.3	Results	54
5	Demonstrator	61
5.1	Problem Overview	61
5.2	Proposed Solution	61
5.2.1	Architecture	62
5.2.2	Enhancement	63
5.2.3	UI and UX	65
5.2.4	Users Feedback	68
6	Conclusions	71
6.1	EO Recommender	71
6.2	EOP-GES Assistant	72
6.3	Potential Developments	74
A	Appendix	85
A.1	Virtual Assistant	85

List of Tables

2.1	Boost Values by Category and Field	24
3.1	Keeped columns per table, primary keys in bold	34
3.2	Performance Metrics of Different Models. <i>"The 'Data' column indicates that 'V' stands for EO Click - Visitors and 'S' stands for EO Click - Sessions. The * specifies if the hyperparameters have been tuned.</i>	44
4.1	Document Usage	55
A.1	Original Questions and Augmented ones. Part 1/4	88
A.2	Original Questions and Augmented ones. Part 2/4	89
A.3	Original Questions and Augmented ones. Part 3/4	90
A.4	Original Questions and Augmented ones. Part 4/4	91
A.5	Employed prompt templates	92

List of Figures

2.1	SBRs publications over the years. <i>Credits to [7]</i>	18
2.2	Earth Online - Homepage	22
2.3	Earth Online - Content Page	23
2.4	Featured from different pages visited at the same time	29
3.1	NNI Dashboard	43
4.1	Image by @chroma	48
4.2	Caption	51
4.3	Execution time by different models	56
4.4	Average Length Ratio by original question	57
4.5	Average Similarity by original question	58
5.1	High Level Architecture of the Demonstrator.	62
5.2	Demo Home Page.	65
5.3	Recommender Demo	66
5.4	Virtual Assistant Demo	68
6.1	Example of information to provide	73
6.2	Example of navigation to get a related page.	74
6.3	First sketch of Earth Online Assistant	75
6.4	Example of individualized assistance.	76
A.1	Assistant Testing grouped by model	86
A.2	Assistant Testing grouped by question	87

Chapter 1

Introduction

1.1 Problem Overview

Accessing ESA (European Space Agency) resources on the web presents a series of challenges, involving various authenticators and resource-specific requirements. This complexity results in a fragmented user experience, leaving users without a clear pathway to access the datasets they require. To address these challenges, Earth Online was introduced as a unified entry point, providing access to articles, datasets, and tools, complete with explicit instructions on accessing specific datasets. This initiative, driven by ESA's commitment to improving user experience, marked a significant step forward, but it still faced limitations.

One prominent issue is that users must navigate away from the Earth Online platform to access external resources. Upon reaching these external websites, users often encounter vastly different interfaces, which can lead to confusion and uncertainty. Users may even question whether they have clicked on the correct link. Additionally, users have expressed concerns about Earth Online's limited offering of related content compared to other websites. Presently, the only available content related to users' interests is comprised of keywords and newly published posts.

Even ESA staff encounter difficulties when attempting to access internal information related to contracts and statements of work. Deciphering the meanings of acronyms and definitions poses another challenge. These challenges collectively underscore the need for innovative solutions to enhance the accessibility and usability of ESA's online resources.

1.2 Research Goal

This thesis sets out to investigate the potential integration of current AI technologies into ESA systems, with a specific focus on EOP-GES (Earth Observation Program - Ground Segment) applications such as Earth Online. The overarching objective is to enhance user experience within these systems. Given the breadth of this topic, we choose to concentrate our efforts on two specific areas: recommendation systems and virtual assistants, since we consider them as promising avenues to address the challenges outlined in our problem overview.

The research agenda encompasses several key phases. First, we delve into the selection of the most appropriate algorithms for implementation, as well as the choice of datasets to underpin our AI-driven solutions. Subsequently, we undertake a comprehensive benchmarking process to evaluate the performance of these models. The subsequent analysis of results forms the basis for the generation of detailed reports and insights, shedding light on the feasibility and efficacy of integrating AI technologies into ESA systems. Through these investigations, we aim to provide valuable contributions to the ongoing efforts to enhance user experience and accessibility in Earth Online and similar applications within the EOP-GES domain.

1.3 Proposal Overview

This thesis, reflecting the culmination of the internship at ESA - ESRIN, encompasses two distinct projects complemented by a demonstrator aimed at showcasing their practical implications. The structure of this thesis is organized into several chapters, each contributing to a comprehensive understanding of the work undertaken.

The journey begins with the "Related Works" chapter, which sets the stage by providing insights into the academic background and existing solutions relevant to our research. It lays the foundation for our exploration of AI-driven solutions in ESA applications.

The "EO Recommender" chapter delves into the first project, the recommendation system. Here, we traverse the entire project lifecycle, from the creation of the dataset to the selection of algorithms, benchmarking, and hyperparameter tuning. This chapter offers a detailed account of our efforts to enhance user experience through personalized recommendations within Earth Online.

In the "EOP-GES Assistant" chapter, we shift our focus to the offline virtual assistant project. This segment chronicles the experimentation phase, beginning with an open call for participation and encompassing prompt engineering and performance analysis. We embark on this journey to assess the feasibility and applicability of such a system within the ESA domain, all while adhering to stringent security and privacy requirements.

The "Demonstrator" chapter captures the challenge of interfacing and presenting our work to stakeholders. To address this, we develop a web application using a modern microservices approach. This chapter offers a glimpse into a real-world use-case scenario, illustrating the potential impact and applicability of our two projects.

The "Conclusions" chapter serves as the culmination of our research efforts. Here, we critically analyze the results obtained in both projects and propose potential applications that could significantly enhance user experience within ESA applications.

Finally, the "Appendix" section provides supplementary tables and figures, offering additional context and detail to augment our research and results.

Chapter 2

Related Work

2.1 Background

2.1.1 Recommender Systems

Recommender Systems (RSs) have become indispensable tools for enhancing decision-making processes in various aspects of daily life, such as work, business operations, education, entertainment, and socialization. With the proliferation of digital content, products, and services, RSs play a crucial role in helping users make more informed and efficient choices. Traditional RSs can be divided in:

- **Content-based Filtering:** Recommends items based on the similarity of their attributes or content to users' preferences. It focuses on matching item features with user profiles. [1]
- **Collaborative Filtering:** Relies on user-item interactions to generate recommendations. It identifies patterns and preferences by analyzing user behaviors and interactions. [2, 3]
- **Hybrid Filtering:** Combines content-based and collaborative filtering to provide better recommendations. [4]

However, traditional RSs have limitations as they often consider all historical user-item interactions equally important in learning users' long-term and static preferences for items.

Indeed, user's item choices depend not only on their long-term preferences, but also on their short-term ones and contextual factors, such as recently viewed or purchased items. The user's short-term preferences are reflected

in their most recent interactions, which usually constitute a small proportion of their entire interaction history [5].

Moreover, the decision on what kind of filtering method to use depends also by the data that will be used for the recommendations and the specific task details. Talking about the data, session or sequence data usage differentiate two kinds of RSs: Session-Based Recommender Systems (SBRs) and Sequence-Based Recommender Systems (SRSs) [6], as they operate on session data and sequence data respectively. A SBRs predicts the unknown part of a session or future sessions by learning co-occurrence-based dependencies within sessions, which can be sequential or non-sequential. In contrast, a SRS predicts successive elements based on the sequential dependencies among them, often utilizing deep learning techniques.

In a recent overview on RSs [7], a detailed taxonomy of SBRs provide the preferred algorithms to use, categorized per task and kind of data, and a deep analysis on dataset details explains challenges and critical points that could affect recommendations.

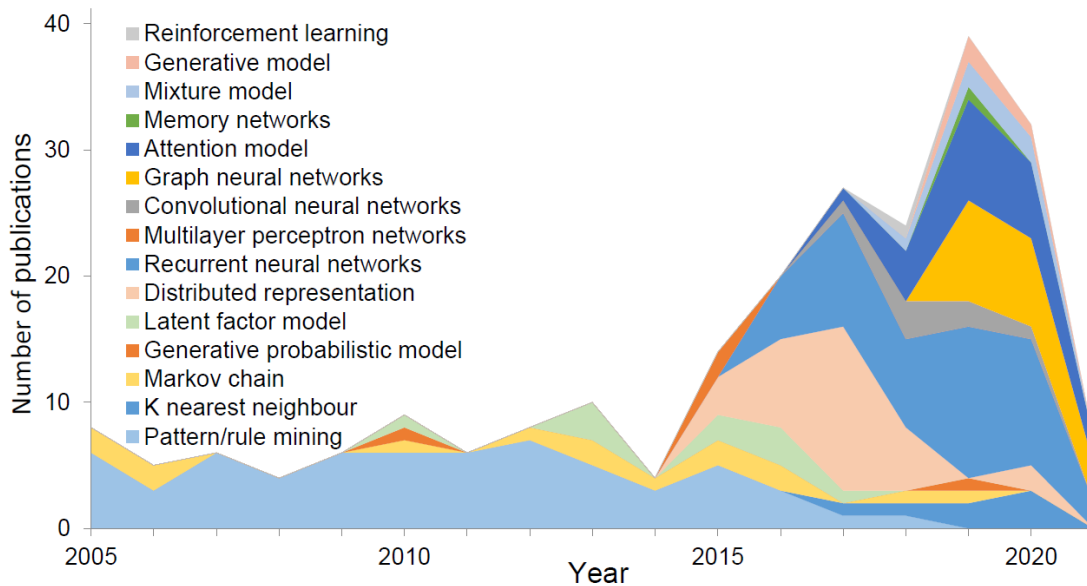


Figure 2.1. SBRs publications over the years. *Credits to [7]*

Despite some researchers did experiments and real tests proving that non-neural approaches perform as well as neural ones in some contexts [8, 9], the interest on neural ones is growing in the last years (Fig. 2.1) because

these approaches find better correlations with unknown and new items with respect of traditional ones. As a matter of fact, non-neural approaches prefer to suggest the most popular items over items that are new, even if they may be more recommendable.

Some of the most remarkable algorithms that have been developed over the years, divided by filtering type, are:

- **Collaborative Filtering:**

- **A2SVD:** A matrix factorization approach that leverages auxiliary data for improved recommendations. [10]
- **BPR:** Bayesian Personalized Ranking that optimizes pairwise ranking to learn user preferences. [11]
- **Caser:** A convolutional sequence embedding recommendation model. [12]
- **GRU4REC:** A recurrent neural network-based approach using Gated Recurrent Units for sequential recommendations. [13]
- **NextItNet:** A next-item recommendation model based on convolutional and self-attention layers. [14]
- **SLI REC:** Sequential Latent Interaction model for session-based recommendations. [10]
- **SUM:** Sequential User Modeling approach for session-based recommendations. [15]

- **Content-based Filtering:**

- **DKN:** Deep Knowledge-Aware Network for content-based recommendations. [16]
- **LightGBM:** Gradient boosting framework used for content-based recommendations. [17]
- **NAML:** Neural Attentive Multi-View Learning for content-based recommendations. [18]
- **TF-IDF:** Term Frequency-Inverse Document Frequency, a classic content-based method.

- **Hybrid Filtering:**

- **Factorization Machines:** Combines linear and matrix factorization models for hybrid recommendations. [19]

- **LightFM**: Hybrid model that integrates both collaborative and content-based approaches. [20]
- **Wide and Deep**: Combines linear models (wide) and deep neural networks (deep) for hybrid recommendations. [21]

2.1.2 Virtual Assistants

Virtual assistants are software applications designed to interact with users through natural language, often via voice commands or text-based chat interfaces. They simulate human conversation and provide automated responses to user queries or tasks. Virtual assistants can perform a wide range of tasks, from answering questions and summarization to text generation and translations.

The concept of virtual assistants dates back to the release of ELIZA in 1966, which employed pattern matching and substitution methodology to simulate conversation. Over the years, various solutions emerged, evolving in sophistication. Examples include Microsoft Clippy (2003), Apple Siri (2011), Amazon Alexa (2014), and Google Assistant (2016), though they were initially limited to basic tasks such as internet searches, smart home device management, calls, and messages. A transformative shift occurred with the emergence of Large Language Models (LLMs), notably GPT [22, 23] and LLaMA [24, 25].

GPT (Generative Pre-trained Transformer) attains remarkable performance through its transformer architecture, featuring a self-attention mechanism capturing local and global context in sequential data. Its effectiveness derives from extensive pre-training on vast text corpora, endowing it with grammar, semantics, and syntax understanding. With a substantial parameter count often in the billions, it grasps intricate language nuances, in addition to its generative nature and contextual awareness that enable coherent text generation.

These models' transfer learning strategies refine their capabilities on specific tasks with smaller datasets, enhancing its versatility, such as in medical applications [26] and coding ones [27]. LLMs' success is attributed to their contextual comprehension, diverse pre-training data, transfer learning, and massive parameter capacity, collectively contributing to its unparalleled language processing prowess.

The remarkable achievements and growing fascination surrounding Large Language Models (LLMs) have also spotlighted a range of important considerations. These encompass concerns related to their environmental impact,

as evidenced by recent discussions on their energy consumption and carbon footprint [28]. Additionally, as LLMs continue to gain prominence, there is a growing apprehension about potential malicious applications that could exploit these models for nefarious purposes [29]. Furthermore, the widespread adoption of LLMs has stirred discussions surrounding the ethical dimensions of their deployment, raising questions about responsible AI usage, bias mitigation, and the potential amplification of societal inequities [30, 31].

Prominent chatbots built on LLMs, like chatGPT [32] and Google Bard [33], provide information, summaries, and translations with a reactivity and ability that position them as formidable contenders against traditional search engines [34].

Beyond the online solutions mentioned, new repositories offer the option to run finetuned models locally, eliminating the need for internet connectivity and enabling private, confidential, and secure implementations.

2.2 Existing solutions

2.2.1 Earth Online

Earth Online (<https://earth.esa.int>) is a website that allows to provide access to a comprehensive range of information related to ESA’s Earth Observation. The portal allows visitors to search and browse content explaining descriptions and technical specifications as well as news, activities, and events focused on the Agency and third-party missions. The portal has a complex navigation mechanism based on global navigation along with search and filtering. The global navigation mechanism allows visitors to navigate through portal landing pages such as for example: Missions, Data, News, Events, Tools.

Earth Online is part of the ESA vision to accomplish the following goals:

- Better illustrate the role and relevance of Earth Observation and ESA EO Missions and Programmes to policy makers, scientists as well as existing and potential value-adders.
- Invite scientists to consider EO data as a natural source for research in many areas and foster new ways to promote the integration of multi-mission data into existing value-adding chains.
- Identify the needs of users who utilise the websites.

- Improve the navigation to all information on the ESA EO websites, making it easy to find with as few clicks as possible.
- Promote existing ESA EO Missions.

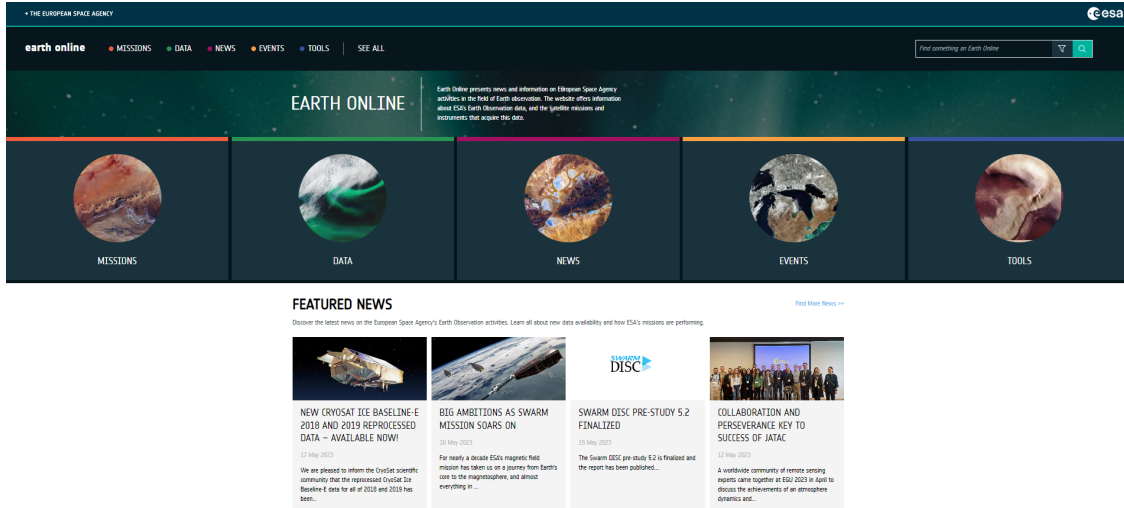


Figure 2.2. Earth Online - Homepage

From a technical point of view, the architecture of the system consists in several servers that aims to provide solutions of high availability and reliability.

The web application is written in Next.js, an open source Javascript framework that enables server-side rendering and static site generation for React applications. It provides important features that allow to create SEO friendly application, light pages for end users and fast security patches thanks to the big community that makes this framework popular.

Earth Online relies on Liferay Portal, a Java-based web platform with features commonly required for the development of websites and portals. Liferay includes a built-in web content management system allowing users to build websites and portals as an assembly of themes, pages, portlets/gadgets and a common navigation.

In the side area of content pages there are the featured posts, that are the newest post in Earth Online and optional contents related to the title and the content of the page, calculated by the index engine.

Matomo

Matomo is the analytics tool used by Earth Online, that is a widely used free and open-source web analytics application designed to track online visits across multiple websites and generate comprehensive reports for analysis. In EO configuration Matomo is capable of tracking user data such as: - country, browser, device, operating system, time spent on a page, referring site to Earth Online, site accessed through internal page links, watched pages and conducted searches.

User persistence in anonymous form is achieved by saving a visitorId and fingerprint, that enables the system to track users for long time that can last even one year.

Matomo offers a comprehensive web application complete with both default and customizable dashboards, enabling analysts to monitor and glean straightforward insights from the logged data. Additionally, it facilitates the establishment of goals, a powerful feature that enables the tracking of website performance aligned with specific business objectives.

Recommendations

Currently there is not a specific recommendation system in Earth Online, so suggestions, keywords and related items are provided by the Search Engine.

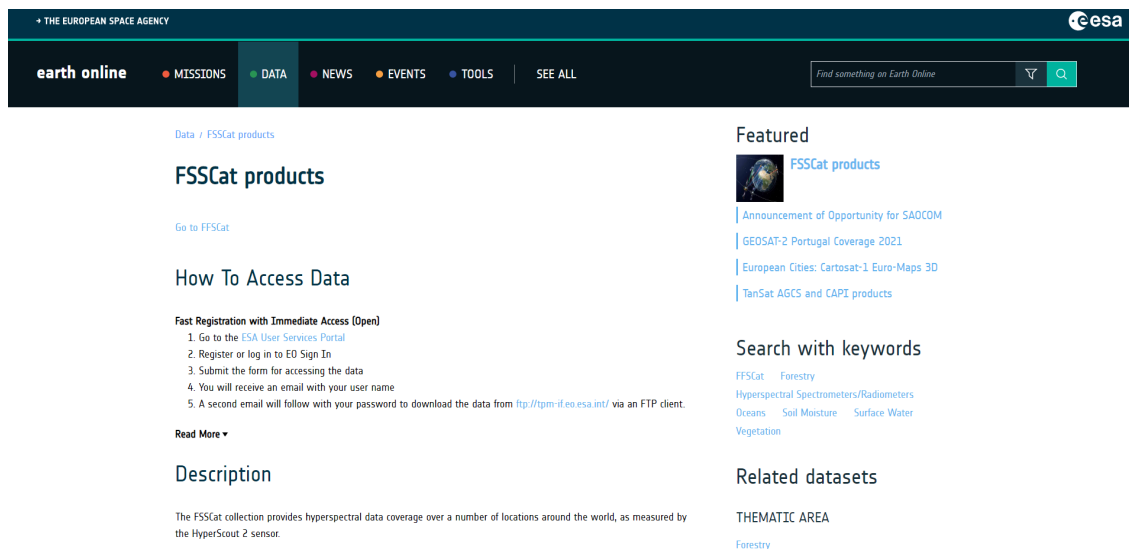


Figure 2.3. Earth Online - Content Page

Earth Online integrates a search engine in order to let the users find the

contents they need with simple text queries and filters for missions, types of instrument, thematic area and finally categories. It’s an implementation of ElasticSearch [35] in which the most important feature is the full-text search that contains aspects like the relevance of the contents (called documents, from this point) to the given query. Thanks to that, results are sorted by their relevance score, in this case the term frequency - inverse document frequency (TF-IDF), that is intended to reflect how important a query is to a document in the collection of documents. The TF-IDF is the product of the two statistics:

- Term frequency $tf(t, d)$, is the relative frequency of term t within document d .
- Inverse document frequency $idf(t, D) = \log \frac{N}{|\{d \in D: t \in d\}|}$, is a measure of how much information the word provides w.r.t. all the documents.

So, the TF-IDF formula is $tfidf(t, d, D) = tf(t, d) \times idf(t, D)$, but in the case of the Earth Online’s solution the score is also normalized by the field-length norm, *the number of words of a document*, for which a term appearing in a short title field carries more weight than the same term appearing in a long content field.

Furthermore the system populates query phrases overall attributes of content stored in the index with different weights called field boosts (Table 2.1) that are used to increase the score to prioritize some contents instead of others.

Table 2.1. Boost Values by Category and Field

Category Name	Field Name	Boost Value
all	title	50
all	description or overview	30
mission	mission objectives	25
mission	mission instruments	20
event	abstract submission	20
event	registration	10

The formula 2.1 is the one used by ElasticSearch to calculate the scores and provide results to an user’s query.

$$score(t, d, D) = \frac{tfidf(t, d, D)}{words\ in\ d} \times boost\ weight(t, d) \quad (2.1)$$

The document analysis is performed by the search engine during the creation or modification of the content and it's composed by two major steps:

1. **Tokenization:** it consists in splitting the block of text into individual terms suitable for use in the search index.
2. **Token Normalization:** it consists in normalize token into a standard form to improve the searchability with techniques like removing the stop-words, html tags and replacing upper-case tokens with the respective lower-case version.

From the perspective of the end user, recommendations are integrated into content pages in different ways. Featured suggestions are presented universally for all types of content, while specific types of content display related keywords and additional metadata. While clicking on featured posts leads directly to the corresponding web page, interactions with keywords or metadata direct users to a search page where contents are filtered by the chosen item.

2.2.2 Virtual Assistants

Currently, there are no active virtual assistants in place. However, given recent advancements and the rising prominence of ChatGPT, the EOP-GES section has become intrigued by its potential. A significant challenge arises due to the nature of company files and information, which may be subject to restrictions. Directly inputting or sharing this data within existing web virtual assistants could potentially breach ESA policies. Therefore, a solution must be devised that ensures strict adherence to ESA policies at every stage of use from uploading documents and framing questions to delivering responses to safeguard compliance with regulations.

privateGPT

PrivateGPT [36] is a tool that allows users to interact privately with their documents using the power of large language models (LLMs). It is designed to be 100% private, with no data leaving the user's execution environment at any point, indeed PrivateGPT can be used to ask questions to documents previously ingested without an internet connection. It is built with:

- **LangChain:** A framework that simplifies working with large language models (LLMs) such as OpenAI GPT4 [23] or Facebook LLama [24]

by providing abstractions for common use cases. LangChain’s use-cases largely overlap with those of language models in general, including document analysis and summarization, chatbots, and code analysis. It includes two modules that allow developers to extend the interactions with LLMs beyond simple chats. The Memory module enables developers to persist state across chains using a variety of solutions ranging from external databases such as Redis and DynamoDB to simply storing the data in memory. The Agents module enables chains to interact with external providers and perform actions based on their responses.

- **GPT4All:** An ecosystem of open-source chatbots trained on a massive collection of clean assistant data including code, stories, and dialogue. It includes several finetuned LLMs models that can be used in PrivateGPT to answer questions without an internet connection.
- **LlamaCpp:** A C++ library for text completion that supports several LLMs. LlamaCpp is used by PrivateGPT to interact with documents and answer questions without an internet connection.
- **Chroma:** A vector store that can be used to store and retrieve vectors. Chroma [37] is used by PrivateGPT to store the embeddings of the documents ingested by the user.
- **SentenceTransformers:** A Python library for computing sentence embeddings that is used by LangChain to load the embeddings model itself.

2.2.3 Limitations and Drawbacks

The present systems effectively deliver the Earth Online services and functionalities as initially specified in ESA’s requests during the development phase with the contractor. However, beyond a year’s span, certain constraints have become evident due to extensive utilization by end-users and performance evaluators. Concurrently, new demands have emerged, prompting a pursuit of innovative solutions. The ensuing points primarily outline the limitations identified through a meticulous examination of the Earth Online system:

- **Inconsistent Featured Area:** Within Earth Online’s content pages, a static featured posts section is located on the side, regularly updated with new posts. Nevertheless, due to the system’s reliance on a cluster, these featured posts are locally cached in each node and that can lead

to variations in featured content across different pages, as depicted in Figure 2.4, where a new post took the place on top and shifted all the others, but with more new contents posted in a short range of time this could lead to completely different featured. This inconsistency introduces incongruent outcomes, as the featured content is ideally expected to be uniform across all pages and users.

- **ElasticSearch’s Scalability Challenge:** The utilization of Elastic Search brings forth substantial benefits in the analysis and querying of documents, leveraging the robust TF-IDF algorithm. However, in the current scenario, where there is a substantial number of pages, its performance falls short of the anticipated standards. Consequently, the process of obtaining desired outcomes for specific queries becomes challenging, as the system struggles to deliver expected results for a given search. The intricacies of handling a sizable volume of data contribute to this discrepancy.
- **Misinterpretation of User Actions:** Matomo classifies user actions into distinct categories such as visits, searches, and downloads. Particularly, search actions offer valuable insights into user intent and can shed light on potential enhancements and user requirements. On the other hand, Matomo’s definition of search actions solely encompasses text queries, excluding filtering queries which are treated merely as page visits. This discrepancy presents a significant challenge, resulting in the search results page being recorded as the most frequently visited page. This issue is compounded by the existing system’s recommendation to employ keywords for content discovery. As a result, Matomo fails to provide meaningful insights beyond indicating user visits to the search page, overlooking crucial information regarding users’ exploration and preferences.
- **Absence of Login Tracking:** Earth Online functions as a website that doesn’t require user logins. Instead, visitor activity is monitored through cookies, that facilitate continuous tracking across sessions and visits, providing valuable long-term insights into website usage patterns. Nevertheless, the straightforwardness of this tracking approach clashes with the reality that deleting browser cookies assigns a fresh identifier to the visitor. This division effectively fragments a genuine user’s journey into distinct digital visitor paths, diminishing the ability to gain a comprehensive understanding of behaviors exhibited by long-term users.

- **Limited Insights from Matomo Dashboards:** The existing Matomo dashboards furnish data concerning top-viewed pages, user navigation paths, and offer statistics on returning and new visitors. However, the prevailing definition of a "returning visitor" solely encompasses those who have previously visited the site and agreed to be tracked through cookies. This classification proves inadequate and overly broad for generating precise analytics, because a "returning visitor" could range from sporadic visits with minimal engagement during a week to regular, daily users of the platform. Merging such disparate data sources results in approximative and unrealistic statistics that poorly reflect user engagement and genuine platform utilization. Additionally, these limitations further contribute to suboptimal analysis capabilities.
- **Limited General-Purpose User Experience (UX):** While the extensive range of keywords, categories, thematic areas, missions, and instruments empowers users with precise filtering options, it can inadvertently overwhelm those unfamiliar with the website's structure.

A pivotal challenge stems from the fact that the search page serves as the nexus of the site, redirecting users from various pages, including keyword clicks. This creates a multifaceted navigation process that can prove cumbersome for users, requiring multiple clicks before attaining their desired content. Differences exist between navigating through mission pages and search pages, contributing to user frustration. To illustrate, consider a user seeking mission-related content. Initially, they will click through mission paths, encountering a different visual layout and content compared to the search page with mission filters. This disparity may lead users to believe they are incorrectly navigating the platform.

Furthermore, when users access specific content pages on a certain topic, the featured posts may not align with their search, necessitating additional clicks on keywords to access related content. This, in turn, leads them to the search page, where they must then sift through results to locate their desired information before finally clicking on it. This protracted interaction necessitates user effort and might discourage them from pursuing their intended goal.

2.2.4 Justification for New Approaches

ESA actively fosters a culture of continuous innovation and embraces emerging technologies, providing ample room for research and cross-disciplinary



Figure 2.4. Featured from different pages visited at the same time

interactions. This environment facilitates the exchange of ideas and collaborative brainstorming among individuals from diverse domains. Furthermore, EOP-GES is currently undergoing a substantial architectural update to introduce novel solutions tailored for new contracts. These enhancements seek to improve efficiency, streamline costs, and more effectively align with evolving requirements and user preferences.

Internship experience within a company boasting an extensive history of managing numerous concurrent projects and contracts further reinforced the need of a tool that facilitate the learning curve for newcomers, a task that aligns seamlessly with the organization’s commitment to innovation and technological advancement.

In addition, the core aspiration of enhancing and simplifying the end-user experience presents an ever-evolving challenge. Despite significant strides in this direction, the journey towards aligning with user needs remains ongoing. These driving forces, combined with the aforementioned limitations, serve as the catalysts for the novel solutions and approaches proposed within this thesis.

Chapter 3

EO Recommender

3.1 Dataset

The foundation of a proficient recommendation system hinges upon meticulous data preparation. We start with a discerning choice, opting for Earth Online (EO) logs as the dataset’s bedrock.

By employing a structured pipeline of data cleaning and preparation functions, we transform the original data creating two distinct datasets that capture the history of clicks at two different levels of granularity: EO Clicks - Visitors and EO Clicks - Sessions. The pipeline encompasses a range of filters and purposeful operations that are applied to the data, ensuring the extraction of meaningful insights and the generation of high-quality training, validation, and test datasets.

3.1.1 Data Source

Various datasets are available online for recommendation systems, offering insights into most common data structures and diverse recommender system methodologies. To fulfill the objectives of the internship project, we need to develop a new dataset from two potential sources: Matomo logs and TellUS tables. As previously discussed, Matomo logs are part of the Earth Online system, while TellUS tables record data access and download requests, including those originating from Earth Online outlinks.

Unlike Matomo logs, the Tellus logs rely on logged users, implying that combining these sources could yield valuable insights similar to how e-commerce systems track users’ preferences. However, connecting these sources is challenging, due to the fact that they are not supposed to do it. While both

sources share certain attributes like IP addresses, country, and timestamp for each entry, linking them directly was unfeasible. These shared attributes have problems from both sides: Matomo's IP addresses are pseudo-random, offering a broad approximation of user location, whereas the monitored Telius tables have missing data for IP addresses and other columns critical for user traceability.

Consequently, we decide to use only Earth Online logs for creating our datasets, because the most important information are contained in this dataset.

Before proceeding with the subsequent stages, it's imperative to underscore the key attributes of Earth Online's (Matomo) logs:

- **Time Coverage:** The logs encompass data from a single year due to GDPR compliance. Entries exceeding a year's time span are deleted on a daily basis.
- **Visitor type:** Visitors are categorized into two distinct types: "New Visitors" and "Returning Visitors." The former denotes individuals who are visiting the website for the first time, thereby possessing a fresh visitor identifier. The latter category represents those who have previously visited the website.
- **Visit (Session):** The concept of a visit is essential. A new visit is recorded by Matomo when a visitor enters the website for the first time or when they perform an action (such as clicking a link or interacting with the site) over 30 minutes after their previous action. This critical distinction signifies that each log entry represents a visit, a term interchangeably referred to as a "session" in this thesis.
- **Interaction type:** Every logged user interaction on the website is classified into one of the following categories:
 - **Action:** Refers to a user clicking on a link, subsequently opening a page on the website.
 - **Search:** Encompasses instances where a visitor triggers the search function by submitting a query in the search bar, leading them to the search results page.
 - **Outlink:** Indicates instances where a visitor clicks on a link redirecting them to an external webpage.
 - **Download:** Represents scenarios where a visitor clicks on a link prompting the download of a file.

- **Goal:** Pertains to interactions triggering predefined goals as configured in Matomo settings.

We decide to consider only interactions of the "action" type, so the term "action" will also be used interchangeably to denote an interaction.

3.1.2 Data Cleaning

The first challenge we encounter while working with Matomo logs is related to the download process. The size of the exported CSV files make it practically impossible to download the entire dataset at once. To address this challenge, we use an approach that involves the segmented downloading of data on a weekly basis. Subsequently, we remove the unnecessary columns, streamlining the logs to ensure readability and reduced file size without compromising the dataset's informative value.

Originally, the log entries encompass a plethora of columns for each session, including visitor-specific details like visitorId, fingerprint, geolocation, and timestamps for the first and last interactions. Each action in the session contributed even more columns such as action type (visit, download, search), time spent on pages, page titles, and URLs. This results in a staggering total of 86 columns characterizing the visitor and session, with an additional 24 columns per action. The presence of multiple actions within a single visit exponentially inflated the column count.

In the initial data cleaning process, we remove the superfluous information, particularly device and browser related details. Then we restructure the data into two distinct tables, as detailed in Table 3.1:

- **Common Table:** Houses visitor and session information with aggregated data.
- **Action Table:** Holds specifics of individual actions within a session.

The cleaned data version, when compared to the original, exhibits a remarkable reduction in size, making it more manageable and potentially feasible for storage and further analysis beyond the Matomo environment.

Finally, we operate additional cleaning steps:

- **Elimination of no-EO domain Url entries:** Interactions with URLs pointing to IP-based websites and the ones without EO domain were removed from the dataset. Also all the interactions with distinct protocols from HTTP(s) were removed.

Table	Columns
Common	idVisit , visitIp, visitorId, fingerprint, lastActionTimestamp, firstActionTimestamp, visitorType, visitCount, daysSinceFirstVisit, visitDuration, searches, actions, interactions, referrerType, continent, countryCode, city
Action	idVisit , actionNumber , pageIdAction, pageLoadTimeMilliseconds, pageTitle, siteSearchCategory, siteSearchKeyword, subtitle, timeSpent, timestamp, type, url

Table 3.1. Kept columns per table, primary keys in bold

- **Interaction Type filtering:** Only interactions of type "action" were retained, as they align with the task’s objective, signifying visitor clicks on specific links or page openings. Interaction types like "outlink" or "goal" led visitors away from Earth Online, while "search" interactions were related to search pages but not the primary focus.
- **Exclusion of Non-Content Pages:** Interactions tied to home or introductory pages, such as mission pages, were excluded from the dataset since the goal was to suggest content pages.

These meticulous cleaning steps are pivotal in refining the dataset and ensuring that our subsequent analysis would yield meaningful insights aligned with the project’s objectives.

3.1.3 Data Preparation

In datasets used for recommendation tasks, items usually have unique identifiers and categories. Depending on the domain, additional features are included like in news recommendation, where typically keywords, abstracts and publishing date are present. These features aid in understanding user preferences and adapting suggestions based on their choices.

Unfortunately, the current dataset lacks such features. Each item only contains the page title and URL. Despite efforts, we can’t obtaining official additional features from other sources. This absence of features presents challenges, including the inability to ascertain if a given URL points to an active page or not. A comprehensive set of available web pages with their

attributes would have been invaluable in addressing this issue and improving dataset reliability.

Feature Engineering

Given the absence of item-specific features, we derive the **category** feature from the URLs using a static schema, as outlined below. For instance, consider the URL <https://earth.esa.int/eogateway/news/an-overview-of-the-aeolus-mission>:

- **Protocol (https)**: This part of the URL specifies the protocol being used, which in this case is "https" (Hypertext Transfer Protocol Secure).
- **Domain/Subdomain (earth.esa.int)**: The domain/subdomain represents the location of the website. In this case, "earth.esa.int" is the domain/subdomain of the European Space Agency's (ESA) Earth Online Gateway.
- **Website Starting Point (eogateway)**: This element indicates that the entire URL path pertains to the Earth Online Gateway website. It serves as a foundational reference point within the website's structure.
- **Category (news)**: Denoting the content's classification within the Earth Online Gateway, "news" specifies the category of news articles. It forms a subset dedicated to news-related content.
- **Page Title (an-overview-of-the-aeolus-mission)**: This portion designates the precise title of the news article. It identifies the distinct content page within the "news" category, providing a human-readable and descriptive summary of the article's content.

By structuring the category feature in this manner, we offer a systematic approach for categorizing and distinguishing different content types within the Earth Online Gateway.

Session Length

Visitors' sessions can exhibit a wide spectrum of durations, prompting the categorization of sessions into three distinct groups based on the number of interactions (n) [7]:

- **Short Sessions** ($n < 4$): These sessions encompass a limited number of interactions, making it challenging to derive substantial context from the interactions. This category also includes sessions with only one or two interactions, representing the extreme cases.
- **Medium Sessions** ($4 \leq n \leq 20$): Medium-length sessions strike a balance between too few and too many interactions. They are less prone to containing an excessive number of irrelevant interactions while still providing essential contextual information suitable for SBRs.
- **Long Sessions** ($n > 20$): Long sessions possess the potential to offer deeper contextual insight for more refined recommendations. However, due to the inherent unpredictability of user behavior, they are more susceptible to incorporating random interactions that introduce noise and subsequently diminish the accuracy of recommendations.

For the purpose of this recommendation system, we focus on medium length sessions, as they present a favorable compromise. These sessions are characterized by their capacity to provide ample contextual information, while their duration remains within a range that mitigates the likelihood of introducing random interactions that could undermine recommendation quality [38].

Item and Category Support

Item support involves assessing the frequency of user engagement with specific items present in the dataset. The measurement of item support captures the extent to which a particular item is chosen, viewed, or interacted with by users.

Complementary to item support, *category support* involves assessing the prevalence of specific item categories within the dataset. This broader metric provides insight into the collective interest in categories, allowing for a holistic understanding of user preferences.

In the landscape of recommendation systems, the establishment of minimum support values holds paramount significance. These thresholds address the challenge of recommending items or categories with minimal user interaction. By enforcing minimum support levels, the system ensures that only items and categories exceeding a certain level of user engagement are considered for recommendations. This practice optimizes resource allocation and prevents the recommendation of items with negligible user relevance.

For the purpose of this recommendation system, we decide a minimum for both item and category support:

- **Minimum Item Support of 1:** With this minimum value, the emphasis is on recognizing the intrinsic value of even the least frequent interactions. This approach ensures that every interaction, irrespective of its rarity, contributes to the understanding of user preferences and behaviors.
- **Minimum Category Support of 10:** This minimum value reflects an intention to introduce users to a spectrum of interests, while also maintaining the significance of recommended categories. Categories with higher support values signify a threshold of collective interest, encapsulating both popular and niche tastes.

Moreover, these chosen support values align seamlessly with operational efficiency considerations. The minimum item support value that we choose safeguards against the exclusion of potentially valuable recommendations, ensuring that the system captures a holistic understanding of user interactions. Simultaneously, the minimum category value strikes a harmonious equilibrium between encouraging diversity and optimizing resource allocation. This ensures that the system focuses computational resources on categories resonating with a substantial user base.

3.1.4 Dataset Creation

At this stage of the project, upon examining the data, we think that there were different way to use data for the creation of the dataset, indeed we can use either the entire historical click record for each visitor or histories confined to individual sessions. So we decide to consider the model’s sensitivity to such changes and whether the dataset’s performance could benefit from either fewer entries with more comprehensive click histories or more entries with shorter histories, leading us to delve into both scenarios and inspiring a more thorough analysis and innovative ideas for potential enhancements.

We choose a specific data format for the dataset, following other examples [39, 40]:

```
<label> <user_id> <item_id> <category_id> <timestamp>  
<history_item_ids> <history_category_ids> <history_timestamp>
```

where columns are separated by `\t`. `item_id` and `category_id` denote the target item and category, which means that for this instance, the objective is

to predict whether the user identified by `user_id` will interact with `item_id` at `timestamp`. `<history_*>` columns record the user behavior list up to `<timestamp>`, elements are separated by commas. `<label>` is a binary value with 1 for positive instances and 0 for negative instances.

It's important to note that the `<user_id>` corresponds to the visitor identifier in one dataset configuration and to the session identifier (originally labeled as `idVisit` in the Common Table 3.1) in the other.

We call the two distinct dataset: **EO Click - Visitors** and **EO Click - Sessions**.

Train, Validation and Test

We partition the dataset to mirror real user behavior while providing a solid foundation for evaluating the model's performance. This approach aims to strike a balance between authenticity and reliability in the context of recommendation systems.

We split the dataset in train, validation, and test sets to capture the temporal progression of user interactions. The validation set encompasses interactions leading up to the final action, instead of test set that contains every last action of the history. All the previous actions are part of the training set, emulating the scenario where the model predicts user preferences without knowledge of their eventual choice. This design mitigates the risk of data leakage and maintained the model's ability to make predictions based solely on historical behavior in the test set. This temporal structure closely resembles how users engage with content over time, ensuring that the model's evaluation aligns with real-world dynamics.

The training set's size allows the model to learn from a broad spectrum of user behaviors, while the validation and test sets presented distinct temporal contexts for rigorous evaluation. This comprehensive segmentation of interactions aims to uphold the authenticity of user behavior patterns and enable effective model training and assessment, fostering a robust recommendation system.

User Unfolding

Beginning with each original line in the dataset, we unfold them into a more granular representation. This process centers around the idea of dissecting a user's historical interactions into incremental stages, revealing the evolution

of their preferences over time. To illustrate, consider a user’s behavior sequence encompassing interactions with various items, each linked to a timestamp. The unfolding commences with the initial step, focusing solely on the first item and its corresponding timestamp. Subsequent iterations progressively integrate additional interactions and timestamps, mirroring the user’s changing behavior. For instance, the initial step captures the user’s engagement with the first item and its timestamp. As the unfolding advances, the second item and its associated timestamp become part of the sequence. This iterative progression persists until the complete sequence of items and timestamps is reconstructed. The outcome is a series of distinct lines, each representing a discrete training instance. These lines effectively capture distinct phases of the user’s interaction history, supplying the model with a comprehensive understanding of their evolving preferences over time.

Negative Instances

EO Clicks - Visitors and Sessions and their train, validation and test sets exclusively consist of real clicks that can be defined as positive instances, necessitating the generation of negative instances, pages not clicked in that sequence. Specifically, we create four negative instances for each positive instance. Negative instance items are chosen randomly from the most popular items, ensuring that they do not include any positive items for each instance and that each negative instance is distinct from the others. The generation of negative instances can help in different aspect, indeed it can be helpful for mitigating overfitting since the model is trained to avoid assigning high probabilities to all observed instances, whether they are relevant or not. Moreover, in this way the dataset is representative of real-world scenarios where true positive instances are relatively rare compared to the vast number of potential negative instances.

3.2 Recommendation System

Within the expansive spectrum of recommendation models, we identify sequential models as the most promising candidates for the task of recommending top-k pages to Earth Online users. Through the utilization of robust Microsoft libraries, we conduct comprehensive benchmarking and fine-tuning of these chosen models, yielding invaluable insights into the datasets and models deemed suitable for adoption. This endeavor establishes a robust

foundation and structured framework for prospective applications within the Earth Online domain.

3.2.1 Exploring Recommenders

In the realm of recommendation systems, **employing sequential models offers a compelling approach** to predict the top K pages that a user is likely to click, based on their prior sequence of interactions. These models excel at capturing temporal dynamics, making them adept at tracking evolving user preferences over time. This translates to more accurate and engaging recommendations. Particularly suited for session-based data, they effectively capture users' intent within individual sessions. Operating on implicit feedback such as clicks or views, they prove advantageous in cases where explicit ratings are sparse. By considering item categories and timestamps, they enhance context-aware recommendations, resulting in more relevant suggestions.

However, certain drawbacks accompany these advantages. The cold start problem becomes pronounced for new users or items without substantial interaction history, affecting the initial recommendations. Data sparsity also presents challenges, as insufficient interaction data for some users hampers accurate pattern recognition [41]. Additionally, these models can be computationally intensive, potentially impacting real-time recommendations and scalability.

To facilitate the implementation and development of the recommendation system, we use a Virtual Machine with the following specifications: *Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz 4-Core, 16GB RAM DDR4*. The framework we choose for building the models is Tensorflow, and we take models' implementation from the Recommenders [42] repository, that provides a rich collection of Jupyter notebooks, focusing on utilizing Python as the primary programming language. This repository covers a diverse array of modern algorithms, encompassing collaborative filtering, content-based filtering, hybrid models, and sequential-based techniques. It includes also examples for data preprocessing, model creation, algorithm assessment, hyperparameter optimization.

3.2.2 Model Selection

We conduct a comprehensive analysis of model performance for which we train multiple recommendation models and then we evaluate them on the

EO Click - Visitors and EO Click - Sessions datasets, as elaborated in the preceding section 3.1.4. Following the evaluation, we select the two best-performing models and perform hyperparameter tuning on them to ascertain whether performance improvements are attainable beyond the default algorithm parameters.

The array of employed models encompasses a diverse range of recommender system approaches, including ASVD, Caser, Gru4Rec, NextItNet, SASRec [43], SLI Rec, SSE-PT [44], and SUM. This collection ensures a comprehensive exploration of various recommendation strategies.

We execute the benchmarking and model comparison using a selection of metrics to offer an all-encompassing evaluation. These metrics include:

- *AUC (Area Under the ROC Curve)*: Gauges the model’s ability to discriminate between positive and negative samples.
- *Log loss*: Assesses model performance based on the log loss of predicted probabilities.
- *RMSE (Root Mean Squared Error)*: Measures the disparity between predicted and actual ratings.
- *Accuracy*: Determines the proportion of accurate predictions.
- *F1 score*: Evaluates the balance between precision and recall.
- *Mean MRR (Mean Reciprocal Rank)*: Calculates the average rank of the first relevant item.
- *Group AUC*: Assesses the model’s ability to differentiate between positive and negative samples within each group.
- *NDCG@k (Normalized Discounted Cumulative Gain)*: Evaluates the relevance of the top k recommended items.
- *Hit@k*: Determines the proportion of recommended items that are relevant in the top k.

Given the diverse scales of these metrics and the inherent challenge of balancing them, we compute a composite score to evaluate models. This score is a weighted and normalized mean, with particular emphasis placed on NDCG@k and Hit@k metrics when $K \neq 4$ —a value previously identified as optimal for real-world applications. This composite score provides a

consolidated representation of model performance, enabling a comprehensive assessment and comparison across a multitude of evaluation criteria.

We employ Early Stopping technique to all training processes, that involves specifying a predetermined value, denoted as N epochs, *in this work* $N = 10$. If the validation metric doesn't improve the best recorded metric for a consecutive span of N epochs, the training process is promptly terminated. This approach proves beneficial in avoiding overfitting tendencies and concurrently trimming down the overall training time.

3.2.3 Hyper-Parameter Tuning

Hyper-Parameter Tuning is a pivotal step in striving for optimal results within a specific context, as the hyperparameters that yield the best outcomes in one dataset test may not be universally applicable. Although we can't tune all the models, due to time constraints, this approach remains pragmatic, acknowledging that real-world application demands extend beyond achieving absolute peak performance.

We concentrate tuning efforts wereon the most influential hyperparameters for the considered models to improve the selected primary metric: AUC. This comprehensive search encompasses a wide array of parameter values including method and standard deviation value for model parameters initializing, L1 regularization for embedding and hidden parameters, L2 regularization for embedding and hidden parameteres, learning rate, loss function, optimizer, training epochs, and batch size.

AUC stands as the preferred primary metric due to its robustness against class imbalance. ROC curves, from which AUC is derived, exhibit insensitivity to shifts in class distribution. Empirical evidence has affirmed that altering the ratio of positive to negative instances within a test set does not impact ROC curves [45]. This inherent property underscores AUC's ability to effectively assess a model's accuracy in discriminating between relevant and non-relevant items in the context of recommendation systems.

We choose Tree-Structured Parzen Estimator (TPE) [46] as tuner, is a specialized algorithm tailored for optimizing quantization hyperparameters. Its iterative operation relies on the historical performance data of evaluated hyperparameter configurations to construct a probabilistic model. This model informs the selection of the next set of hyperparameters for evaluation, enabling efficient exploration of the hyperparameter space. TPE's suitability lies in its ability to strike a balance between optimizing parameters within a finite timeframe, but it's worth noting its limitation in discovering intricate

relationships between different hyperparameters.

We use the open-source software NNI[47], developed by Microsoft, to conduct our Hyper Parameter Tuning . This automation tool empowers users to conduct automated machine learning (AutoML) experiments, by dispatching and executing trial jobs generated by tuning algorithms, NNI facilitates the search for optimal neural architectures and/or hyperparameters across various environments, including local machines, remote servers, and cloud infrastructure.

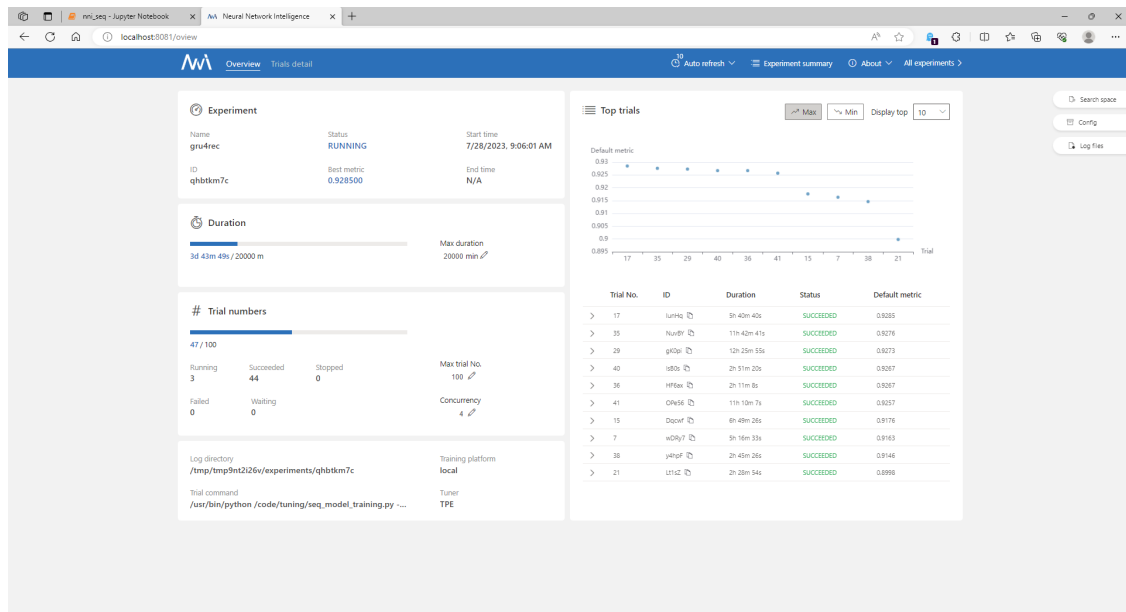


Figure 3.1. NNI Dashboard

Central to the effectiveness of this tool is its seamless integration of metrics monitoring across optimization and trial stages. Each trial represents a test executed with a distinct combination derived from the original search space, allowing for insightful comparisons and decision-making throughout the hyperparameter tuning process.

3.2.4 Results

Following the described procedure best results are shown in the table 3.2.4. From this analysis, SLI Rec and Gru4Rec emerge as the best models for most of the metrics, outperforming all the other tested models and the dataset version that provides the best results is EO Clicks - Visitors.

Data	Algorithm	Train time (min) ↓	Test time (min) ↓	AUC ↑	Logloss ↓	RMSE ↓
V	SLi Rec *	93.1981	0.5078	0.9377	0.2353	0.2585
V	SLi Rec	46.9574	0.3544	0.9300	0.3249	0.3103
V	GRU4Rec *	140.4682	0.4106	0.9290	0.2477	0.2653
V	GRU4Rec	68.5001	0.3249	0.9229	0.3330	0.3165
V	Caser	21.1053	0.2166	0.9146	0.3261	0.3158
S	GRU4Rec *	35.5298	0.7487	0.9260	0.2625	0.2726
S	SLi Rec *	20.7259	0.7919	0.9230	0.2692	0.2796
S	SLi Rec	24.1463	0.6026	0.9175	0.3591	0.3260
S	GRU4Rec	13.3048	0.5541	0.9131	0.4059	0.3521
S	ASVD	17.1794	0.5258	0.9121	0.3590	0.3289

Table 3.2. Performance Metrics of Different Models. "The 'Data' column indicates that 'V' stands for EO Click - Visitors and 'S' stands for EO Click - Sessions. The * specifies if the hyperparameters have been tuned.

Dataset. The preference for the *EO Click - Visitors* over the *EO - Click Sessions* is rooted in its inherent capacity to provide enhanced recommendation system performance. By encompassing a comprehensive history of individual user interactions, the Visitors dataset enables models to discern intricate behavioral patterns, capturing diverse preferences and temporal dynamics. This granularity empowers recommendation systems to offer personalized suggestions that align with real-world scenarios, where user interests go over individual sessions. Moreover, the dataset’s contextual richness facilitates the identification of longer-term trends and evolving preferences, contributing to more accurate and adaptable recommendations.

Algorithm. Among the array of models assessed in the testing, SLi Rec and GRU4Rec emerge as the frontrunners, demonstrating remarkable capabilities in the task of personalized recommendation. Notably, SLi Rec showcases superior performance, probably because SLi Rec’s unique advantage of being time-aware, enabling it to harness temporal patterns and dynamics within user interactions. This temporal awareness empowers SLi Rec to decipher user preferences over extended periods, a facet not fully accessible to the other models. In light of the metrics reported, including a AUC of 0.9377, a log loss of 0.2353, a mean MRR of 0.9003, and a group AUC of 0.9291, SLi Rec establishes its prowess in distinguishing positive and negative samples, predicting probabilities with precision, ranking relevant items adeptly, and

catering to diverse user groups effectively.

Hyper-Parameter Tuning. Hyper Parameter Tuning significantly enhances GRU4Rec and SLi Rec models. Log loss drops by 27%, indicating better distinction between interactions. RMSE lowers by 16%, improving interaction pattern approximation. Tuning aligns models with data, boosting accuracy and personalized recommendations, accentuating the practical significance of hyperparameter optimization. However, it's worth noting that the TPE is not exhaustive like Grid Search, leaving room for the possibility of achieving even better results with further optimization or specifying a different primary metric. The prolonged training and test times following hyperparameter tuning can be linked to different values for batch size, loss function, initialization values, and learning rate that could slow the learning process, this in particular can explain why the difference in training time is so high.

Final considerations. There is a real difference between synthetic tests and empirical tests for recommendations and a test with incredible results could lead to a not real one [9].

Furthermore, negative instances are generated and the model may be taught to don't recommend items that could actually be interesting for end users. In synthesis a more detailed dataset and an empirical test could provide better and real results about the best model.

Hyper Parameter Tuning improves results, but it costs execution time and the TPE tuner could have brought to sub-optimal results. Using a more exhaustive tuner like Grid Search and choosing other primary metrics could bring to different optimum even if it's likely that SLi Rec performs better since it has very good performances in all the considered metrics.

The results we obtain offer valuable insights into the choice of the optimal dataset among the two proposed options. Furthermore, this study presents a comprehensive survey of methodologies and essential tools for building effective recommendation systems. This structured project comprises multiple Jupyter notebooks and Python libraries, serving as a solid foundation for future analyses and implementation within the ESA domain.

Chapter 4

EOP-GES Assistant

4.1 Exploring PrivateGPT

PrivateGPT is a widely acclaimed repository on GitHub, amassing a remarkable 37,000 stars at the time of writing, indicative of its popularity within the developer community. The reasons why it's becoming so famous can be related to its key features like the possibility to answer only on provided documents, the fact that it works offline and the possibility to use the Large Language Model (LLM) that users prefer. This opens up exciting possibilities for collaborative implementations, with contributions ranging from translations to the integration of the ChatGPT API.

The workflow of PrivateGPT consists of two main steps:

1. **Ingestion:** During this phase, users are required to provide documents in any supported format to the system. Following the repository's code, documents should be placed in a designated folder. Running the command `python ingest.py` in the terminal initiates a sequence of actions. First, it employs the appropriate parser for each document, and then it stores the resulting embeddings in a Chroma database. This database allows for the storage of embeddings and associated metadata, facilitating document embedding, querying, and retrieval [37].
2. **Question and Answer (Q&A):** This step is where users can pose questions and receive responses. By running the command `python privateGPT.py`, the script loads the model and Chroma database, then users are prompted to input their queries in the terminal. A LangChain RetrievalQA process, based on the database and a specified LLM, is then executed. After the answer is generated, the system provides the

response time and cites a limited number of sources, each containing the filename of the respective document and the relevant text fragment from which the answer was derived.

The search for related embeddings is a pivotal aspect of the system, and its functioning is elucidated in Figure 4.1. The default embedding model utilized is the well-regarded all-MiniLM-L6-v2, renowned for its simplicity and speed [48]. It's a sentence transformer fine-tuned using a concatenation of diverse text datasets.

A standout feature of RetrievalQA by LangChain is its prompt customization, empowering developers to infuse additional context into queries, thereby influencing the answer generation process. Furthermore, the configuration is highly flexible, offering numerous settings to experiment with in the quest to enhance performance.

It's noteworthy that this system defaults to CPU computation, which can affect performance, resulting in relatively slower execution times. Indeed, the recommended model for the repository is GPT4All-J v1.3-groovy, a lightweight language model (LLM) [49] developed by Nomic-AI based on GPT-J [50], a Transformer model similar to GPT-2.

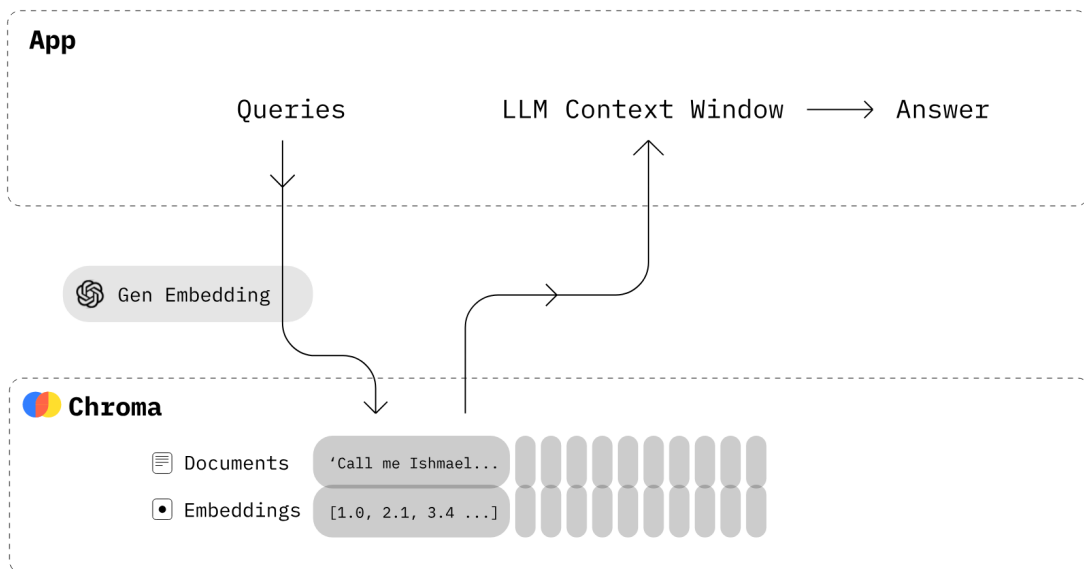


Figure 4.1. Image by ©chroma

4.2 Experimentation

The burgeoning interest in chatGPT has sparked a deep curiosity surrounding privateGPT, primarily due to its offline capabilities. The heightened curiosity prompted several EOP-GES managers to start asking its potential applications in their operations. In response to this keen interest, we initiate to investigate the practical applications of privateGPT and assess its feasibility through an experiment.

The virtual machine (VM) we employ for executing all the scripts boasts the following specifications: Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz with 4 cores, 16GB DDR4 RAM, and an NVIDIA A40 GPU with 12GB VRAM.

Prior to commencing the experiment, we implement a series of optimizations, summarized as follows:

- **GPU Support:** We modify the original code to support GPU calculations. This enhancement significantly improves response times, with a reduction of over 50%. For specific large language models (LLMs), it reduces processing times from 50 minutes to just 3 minutes.
- **A new Sentence Transformer:** While the *all-MiniLM-L6-v2* model is known for its trade off between quality and speed, the focus of this experiment is on content over speed. Therefore, we employ the *all-mpnet-base-v2* model [48] to generate embeddings, offering a higher level of quality.
- **Wide LLMs support** To explore and compare different LLMs, we download additional models from ©HuggingFace [51, 52]. Included models are various quantized versions of Vicuna-13B [53], LLaMA2, LLaMA2-chat [25], and Orca (a fine-tuning of LLaMA2) [54]. All the models used are in their 13B parameters version. It’s not possible to test bigger networks because of hardware limitation; as LLMs are memory-intensive, and model loading is just one part of the process. The adopted models do not exceed 12GB in RAM usage.
- **New script design:** Recognizing the need to ask numerous questions and acknowledging that the current approach requires continuous manual input of questions, we adopt a new approach. The new script we code allows the system to process questions from a specific file, ensuring a seamless flow where the model automatically proceeds to the next

question. This approach enables the system to function in background without requiring constant user interaction.

- **Saving Answers:** In the existing setup, questions are typed into the terminal, and the responses are printed only within the terminal interface. This limitation makes it challenging to analyze responses later or store them in a structured manner. To overcome the problem, we make the system save answers and their associated sources as both CSV and JSON files. Each saved entry includes details such as the currently running model, the question asked, the response generated, the time taken to reply, and the sources consulted.

4.2.1 Methodology

The experimentation process adhere to a structured series of steps, meticulously designed to offer an exhaustive and comprehensive analysis to ESA.

1. Open Call for Participation

To subject the system to a real-world scenario and garner the attention of the Agency, the initial phase of the experiment involves an open call inviting colleagues to partake in the test. The objective is to collect documents and formulate relevant questions based on genuine user requirements. This approach aims to extract insights into the strengths, weaknesses, and limitations of the proposed solution.

This phase do encounter some challenges, primarily due to the majority of the team being on vacation. Nevertheless, participation is secured from one manager and two other colleagues, facilitating the commencement of the experiment.

2. Prompt Engineering

It refers to the process of designing and selecting specific input prompts to guide the output generation of LLMs and in general other generative models that must interpret natural language. It is widely recognized that the phrasing of a question can yield various, often nuanced, responses. For instance, studies have shown that this principle applies not only to text-based models but also to image generation, where slight modifications in input phrasing can result in diverse artistic outputs [55]. Furthermore, ensuring consistency in responses is a challenging aspect of prompt design [56].

Designing effective prompt templates remains a challenge, as it requires a deep understanding of the interplay between the LLM and the prompt [57]. In the context of this particular experimentation, our goal for prompt engineering is primarily exploratory. It aims to illustrate how answers can vary in response to slight variations in the input question. To achieve that, the original query provided by participants serves as a basis for generating four additional versions of the same question by chatGPT using the prompt: *Considering the following question, write other four version of it, with the same meaning, but with different words or order of words: {question}*. An instance of the described process is exemplified in Figure 4.2.1 and all the questions are shown in Appendix starting from Table A.1

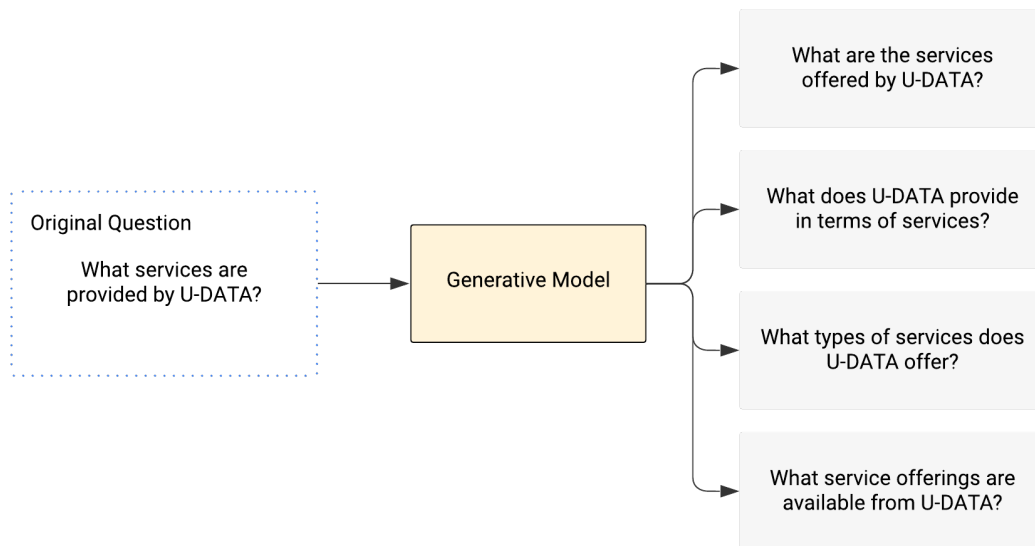


Figure 4.2. Caption

3. Document Analysis

The effectiveness of the model hinges on its ability to furnish accurate information sourced from the provided documents. As part of this experimentation, this phase involves a detailed analysis of which sections of the documents are utilized to generate responses. Additionally, another topic of this step is to analyze how the sources of information vary when the same question is posed within a more extensive collection of documents.

4. Models Comparison

As explained earlier in section 4.2, various models and model versions have been prepared for testing purposes. A pivotal phase of the project involves the comprehensive evaluation of these diverse models. This evaluation entails a comparative analysis of their responses, resource requirements, and execution times. This comparative assessment is a crucial milestone for the project, as it aids in gauging the resource allocation necessary for a real-world application. It is anticipated that more complex models will yield more accurate and satisfactory responses, making this analysis instrumental in decision-making.

4.2.2 Evaluation

The experimentation phase involves the utilization of various documents, each presenting distinct scenarios that represent different use cases. Specifically, the experimentation addresses three use cases:

- **Analyzing Third-Party Offers in Alignment with Agency Needs:** Often, external companies submit proposals for specific contracts, believing that their services align perfectly with the agency’s requirements. However, it falls upon the Agency to assess whether these offers are indeed valid and meet their needs. In this scenario, documents from the third-party company’s presentation and the agency’s own needs are provided for evaluation.
- **Summarizing Lengthy Documents:** In situations where extensive documents require summarization or the extraction of key insights, employing a virtual assistant to perform these tasks can significantly improve efficiency. For this use case, a 300-page document is provided to the system with the task of generating a concise summary.
- **Retrieving Information from Specific Contract Documents:** Managers routinely manage sets of contracts, necessitating the perusal of numerous documents to gain a high-level understanding of ongoing activities. This information is crucial for providing support, requesting additional resources, and handling issues. To address this scenario, documents such as the statement of work, weekly meeting reports, and user feedback related to a specific contract are provided for evaluation.

These diverse use cases enabled a comprehensive assessment of the system’s capabilities and its effectiveness in addressing real-world Agency needs.

Metrics

In this experimentation, a combination of qualitative and quantitative metrics is employed to comprehensively evaluate the outcomes.

Perceived Answer Quality. Both participants and the candidate assess the perceived quality of the answers. We consider factors such as whether the answer lacks vital information, is overly concise, or if the source documents are appropriate for the given question. This qualitative assessment offers valuable insights into the human perspective of answer quality.

Quantitative Metrics. Several quantitative metrics are utilized to enhance the evaluation process, particularly for comparing answers in scenarios involving multiple documents or different models. These metrics are as follows:

- **Answer Length:** This metric measured the length of the answers, providing insights into their conciseness and informativeness. Longer answers generally indicate a more comprehensive response and a higher likelihood of providing valuable information for the given question. An answer length will always be compared to others, indeed it will be called Answer Length Ratio and calculated by the formula $\frac{\text{len}(\text{second answer})}{\text{len}(\text{first answer})}$. It will be averaged also among all the answers within the same original question or models.
- **Answers Similarity:** Cosine similarity is employed to gauge the similarity between two answers. To calculate this, the Sentence Transformer generated embeddings for the answers, and Scikit-learn’s cosine similarity function is applied [58]. This metric helps to assess how similar two answers are in terms of content. For instance, considering only two sentences the similarity is calculated like in Listing 4.1 using Sentence-Transformer [48] Python library.

```
from sentence_transformers import
    SentenceTransformer, util

model = SentenceTransformer('sentence-
    transformers/all-MiniLM-L6-v2')

answer_1 = "hello , how are you?"
answer_2 = "how are you"

embedding_1 = model.encode(answer_1 ,
    convert_to_tensor=True)
embedding_2 = model.encode(answer_2 ,
    convert_to_tensor=True)

sim = util.pytorch_cos_sim(embedding_1 ,
    embedding_2)
```

Listing 4.1. Similarity Algorithm

- **Document Sources Count:** Another quantitative metric is counting how many times a specific document is utilized as a source. This metric aids in understanding which types of documents are considered relevant and which are not, providing insights into the sources' significance.

4.2.3 Results

In the context of analyzing third-party offers, the system encounters challenges in recognizing the third-party company mentioned in the presentation document. This occurs because the name of the company is never explicitly associated with the services they provide within the document. To address this issue, providing more context to the model, such as associating "we" and "our" with the specific company, might be necessary to improve accuracy.

For the task of summarizing large documents, it presents a different set of challenges. While LangChain provides a specific summarization chain, it struggles with very long documents. The summarization process involves reading all the pages, splitting each page into chunks of embeddings, and does not allow specifying the desired length of the summary. To enhance this capability, additional chains or methods for summarization, as suggested by LangChain [59], may be required.

Regarding the use-case involving contract documents, the system receives substantial support from a manager in terms of the number of documents, questions, and assistance. We conduct the evaluation in two rounds, with different document sources, using provided questions and augmented ones listed in Appendix starting from table A.1. In the first round, we upload only requirements and statements of work. In the second round, we add also reports covering various time spans (weekly, monthly, etc.). The document usage patterns differs between the two rounds, with monthly reports being preferred in the later stage, as shown in Table 4.1. This preference is likely because monthly reports contain notable information and constitute the majority of the available documents. Additionally, the Statement of Work (SOW) document, despite being a single document, contains numerous definitions, parts of the requirements, and general contract information, making it a frequently used source.

Type of Document	First Round	Second Round
Statement of Work (SOW)	65	43
Monthly Service Management Report	N/A	71
User Support Service Requirements	55	42
Data Access Service Requirements	63	40
Yearly Quality Assurance Report	N/A	34
User Support Service Monthly Report	N/A	30
Data Access Service Monthly Report	N/A	39
Bimestral User Experience Book	N/A	15
Service Level Requirements	15	0

Table 4.1. Document Usage

From a quantitative perspective, Figure 4.3 illustrates the minimum, average, and maximum execution times required to answer a question. Notably, for the Vicuna models, which have similar complexities but differ in quantization, the average answer times are comparable. An interesting observation is that the heaviest Vicuna model in terms of RAM usage (q5_1) exhibits both the lowest minimum and the highest maximum answer times. In contrast, LLaMA 2 demonstrates a relatively low minimum response time but a higher average response time.

An important point to note is the considerable variance in response times among the original questions and their augmented versions generated by chatGPT, as depicted in the third chart of Figure A.2 in the Appendix. This

variance indicates that response times can significantly fluctuate depending on the specific question and the documents upon which the models rely. However, it’s essential to emphasize that the primary goal of this analysis is not to pinpoint exact execution times but rather to provide proportional comparisons. The hardware we use for testing is not optimized for production, and response times are heavily influenced by these hardware limitations.

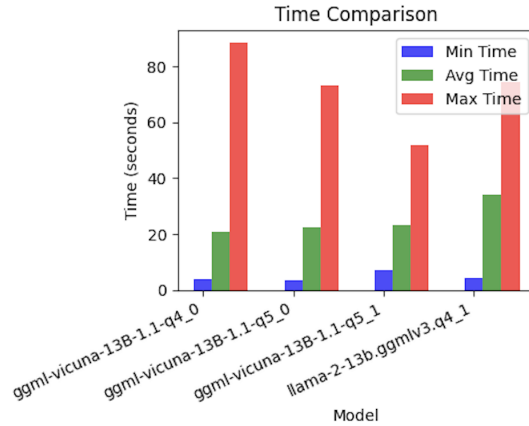


Figure 4.3. Execution time by different models

Discussing the nature of questions and how various models handle both original and augmented versions of questions, Figures 4.4 and 4.5 provide insights through averaged statistics grouped by the original question. These statistics represent the average results across all questions evaluated for all tested models and are then reported for each original question.

We present this data in two scenarios: the first round ($f(F)$), where the document collection does not include periodic reporting documents, and the second round ($f(S)$), where document collection does encompass periodic documents. To provide further context, we calculate a ratio, denoted as $\frac{f(F)}{f(S)}$, which signifies the relationship between the two values.

These analyses allow for an understanding of how the nature of questions and the presence of periodic reporting documents influence the responses generated by various models.

The Average Answer Length, as depicted in Figure 4.4, represents the average fraction of the original question’s answer length compared to the answer lengths of the augmented questions. Therefore, values closer to 1 indicate that the answer lengths are more similar across these questions.

The same principle applies to the cosine similarity, as illustrated in Figure

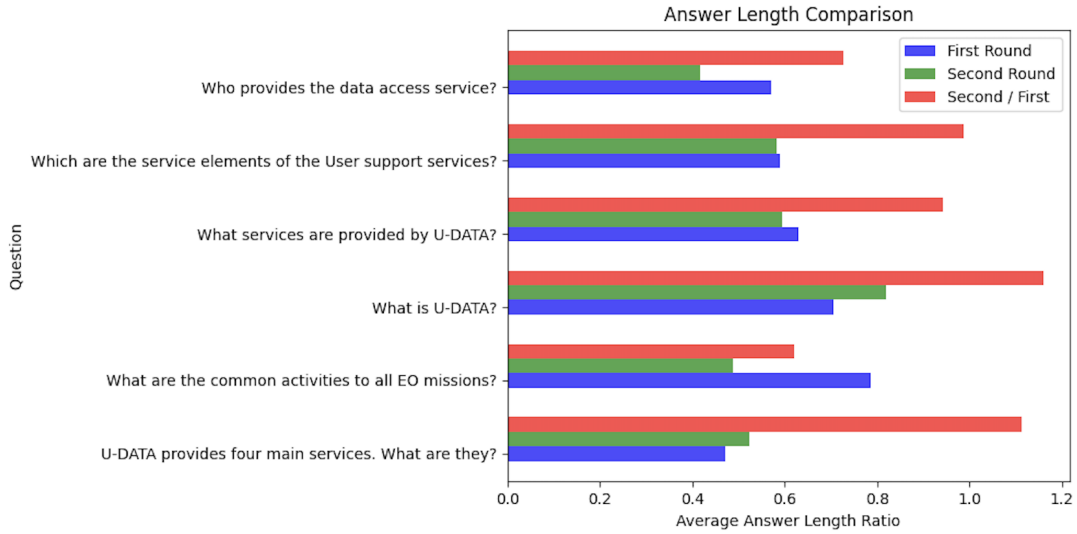


Figure 4.4. Average Length Ratio by original question

4.5. A value closer to 1 indicates that the answers likely convey similar concepts.

Upon examination, it becomes evident that both time to answer and these metrics are significantly influenced by the nature of the question and the sources involved. For the group of questions related to the original question *"What are the common activities to all EO missions?"*, there is substantial variance between the first and second rounds in terms of answer length. This variation may be due to different interpretations of certain questions, resulting in answers that contain more detailed reporting data compared to others that maintain a higher-level and informative response.

Furthermore, questions related to *"Who provides the data access service?"* generate markedly different answers in terms of content, as evident from the low average similarity value. Here, too, the differences in answer content and length may stem from variations in the depth of information provided in the answers.

An interesting and challenging aspect that demands attention is the selection of appropriate prompt templates [60] for the models to provide the best context to elicit the model to provide the most suitable answer. For instance, the prompts that we use for the tested models are shown in Appendix in Table A.1. Interestingly, for Vicuna models, this issue appears to be of lesser importance. In cases where it's impossible to provide an answer, they simply reply so.

Conversely, the LLaMA 2 models, especially the chat version, necessitate more explicit prompts. These prompts need to instruct the model not to generate a response if an answer can't be found in the documents. Additionally, specific instructions are required to make the chatbot more helpful and respectful in its responses. Despite these explicit instructions, there are instances, as exemplified in Listing 4.2, where the model generates a detailed, high-quality answer. However, it erroneously infers the meaning of the acronym "U-DATA" as "User-friendly Data Access," even though the prompt explicitly directs it not to do so. This highlights the nuanced challenges of instructing language models to provide precise and contextually appropriate responses.

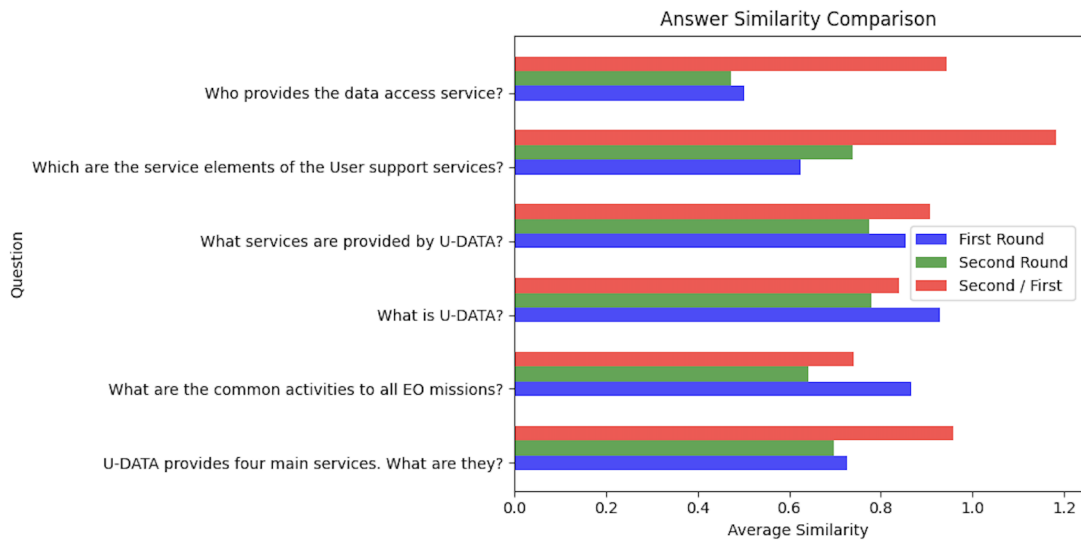


Figure 4.5. Average Similarity by original question

```
"model": "llama-2-13b-chat.ggmlv3.q4_1",
"question": "What is U-DATA?",
"answer": "Hello! U-DATA stands for \"User-friendly
Data Access\" and it's a service provided by the
European Space Agency (ESA) to give users access
to Earth Observation (EO) payload data. The
service includes data access, user support, and
engineering services for managing and maintaining
applications and catalogues for supported
satellite missions. U-DATA is integrated with
other EO PDGS services responsible for web
publishing, data management, data quality, and
scientific support. It also relies on enabling
services like Information and Communication
Technology (ICT) services and software application
maintenance services. Is there anything else you'd
like to know?",
"sources": [
  "Statement of Work.pdf",
  "Monthly Report.pdf",
  "Yearly Quality Assurance Report.pdf"
],
"contents": [ "...", "...", "...", "..." ],
"time": 41.9
```

Listing 4.2. Example of a generated reply

In conclusion, this analysis underscores the importance of carefully designing questions and providing clear guidelines to end-users to ensure consistency in answers generated by language models. The LLaMA models with 13 billion parameters, particularly Vicuna, have shown promising performance. It would be intriguing to evaluate even larger models with 70 billion parameters to determine their suitability for this specific task.

The LLaMA 2 models demonstrate the ability to extract nuanced information from original documents, leading to high-quality answers. However, it's crucial to validate these answers since they may contain a mix of genuine and fake information. This challenge highlights the need for careful scrutiny and verification when relying on such models for critical tasks.

Chapter 5

Demonstrator

5.1 Problem Overview

Once we complete the works on the EO Recommender 3 and the EOPGES Assistant 4, their results provide information in various formats like benchmark tables, answers with details and numerous notebooks and python scripts. Since the exploratory purposes of the internship finalizes with a presentation, just showing these results to an audience of managers with a very poor background in the faced topics looks not enough to justify the time, sweat and work that we put on it.

Moreover, giving just results or the technology itself don't give the idea of what can be done and how it could help internal or external users, as Steve Jobs said *"Some people say, "Give the customers what they want." But that's not my approach. Our job is to figure out what they're going to want before they do."*

5.2 Proposed Solution

To facilitate a clear understanding of how these projects can be applied and how they function, we develop a "Demonstrator". This demonstration platform enables users to create recommendations by selecting a specific user with their history. Users can also interact with a virtual assistant, uploading their documents and asking questions as they desire.

The Demonstrator takes the form of a web application, with dedicated sections for each project. It operates through a series of microservices to simulate the functionality of a real production application and enhance user

engagement.

5.2.1 Architecture

The system comprises five components, each deployed as a container using Docker [61]. Docker, combined with the microservices approach, allows for the removal of strict dependencies and provides the flexibility to replace any of the containers while ensuring the consistency of the API.

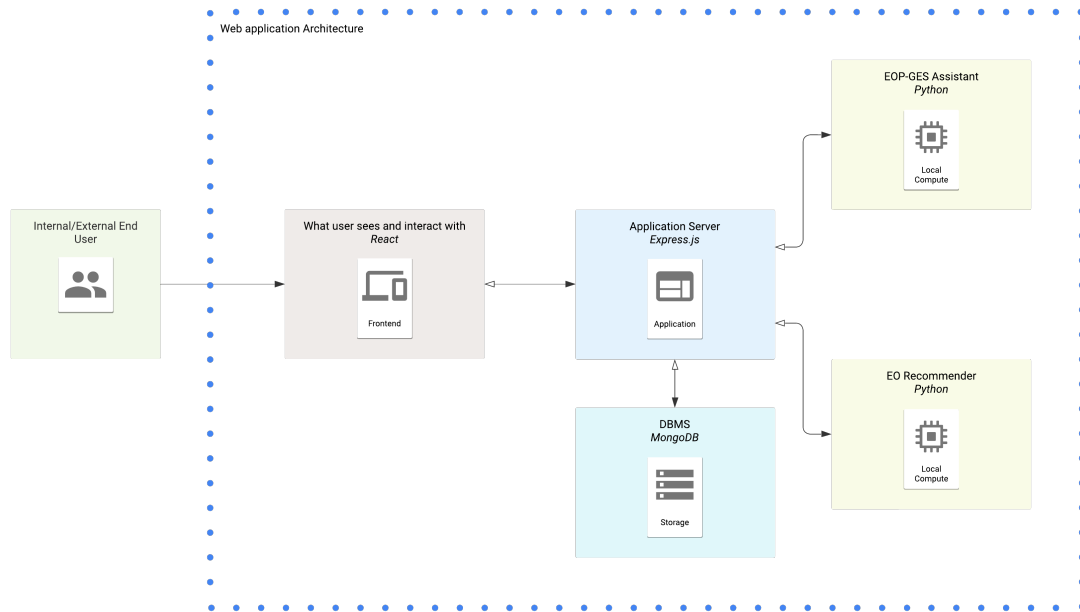


Figure 5.1. High Level Architecture of the Demonstrator.

The architecture we design is showed in Figure 5.1 and it's explained in the following overview of the system's components:

- **Front-end:** This is a React-based [62] application with Bootstrap [63] styling. It consists of three main pages: Homepage, Recommender, and Assistant. All the data displayed on the front-end is retrieved via API requests to the Application Server.
- **Application Server:** This component serves as a REST API server and is developed using the Express.js [64] web application framework. It plays a pivotal role by delivering results from the AI models to the front-end through a unified set of APIs. This separation of concerns eliminates

the front-end’s dependency on AI models, enabling easy model replacements. The Application Server communicates with both the AI models of the projects and a DBMS.

- **DBMS:** MongoDB [65], a NoSQL database, is used in this system to store visitor details for the recommender and to save assistant replies for the assistant component. We decide to adopt a NoSQL database, specifically MongoDB, because of its schema-less nature. This schema-less characteristic allows for flexibility in altering the structure of documents without causing any significant slowdowns or disruptions in the system’s operation. It provides an agile and adaptable data storage solution, which is particularly advantageous when dealing with evolving data requirements and document structures.
- **EO Recommender:** A Flask [66] REST API server that serves as the recommendation engine for specific visitors. This server provides tailored recommendations, including top and flop page suggestions, based on individual user profiles. Its primary function is to respond to requests for recommended pages using a customized version of the code originally developed for the EO Recommender project.
- **EOP-GES Assistant:** A Flask-based REST API server that serves a modified version of privateGPT. The server’s primary purpose is to load the model into memory and respond to questions via REST API requests.

A key point in our architecture is the independent deployment of AI models. Python, while commonly used for AI tasks, is well known for being resource-intensive and time-consuming [67]. Therefore, we employ Python only in the portions of the system where it is necessary, specifically for AI model execution. This approach helps reduce energy consumption and reduce latency.

5.2.2 Enhancement

The original code of the projects is not designed for serving via REST API; it’s initially intended for testing and academic purposes only. To make it works for the demo and enhance its performance and make it more robust, we implement further several adjustments.

- **Refactoring Recommendation System:** The Recommendation System’s code originally consists of just a few lines for model initialization and testing execution. We restructure the code to adhere to Flask guidelines, including the use of annotations. Furthermore, we modify the latter part of the execution. Instead of returning the score for the provided test dataset, the system now uses the score to sort items accordingly, returning only the top and flop pages with details like the url, the category and the score itself.
- **Refactoring privateGPT:** PrivateGPT undergo a complete refactoring. Previously, it only works with blocking user input, and most of the main code is contained within a while loop and the enhancement we make in Section 4.2 is not enough for this stage.

In the refactored code, the selected model is downloaded if not already available and loaded into memory to wait for requests. Only when a request is received, the query part of the code is activated. This change significantly alters the code structure. Indeed, the code is no longer sequential but follows the Flask coding approach with annotations. The main part of the script primarily consists of database initialization and Flask application setup. Another change relates to the ingestion process. Previously handled in a separate script, it is now an accessible function executed within the Flask app.

- **Adding the cache:** We implement a caching system to expedite assistant responses. As previously observed in Figure 4.3, response times can vary significantly. Given that the presentation needs to be live and many questions might be repeated, the concept is to store all answers with their respective questions. This way, if the same question is asked again, the system can provide the existing answer instead of querying the model again. We integrate this approach into the Application Server and it substantially reduces response times for previously asked questions. Furthermore, the caching system takes into account the documents that are ingested. Consequently, if a user uploads new documents, the answer will be regenerated to consider potential new sources and, consequently, provide new answers.

Furthermore, it’s important to note that this improvement is not solely for the demo but should also be integrated if the system is to be deployed. The utilization of a caching system can substantially reduce latency and resource consumption, especially considering that questions are likely to

be repeated or similar in real-world usage.

- **AI Models hot loading:** Another significant improvement we provide is the loading of the models in the memory at start up. In the traditional approach, loading models and potentially documents takes time, which is added to the response time for a request. To eliminate this additional latency, models are now loaded into memory during the startup of the Flask applications. This means that when a question or recommendation is requested, there is no extra delay beyond the actual time required to perform the task.
- **Introducing APIs:** We introduce brand new APIs to facilitate communication between the applications. These APIs adhere to best practices, utilizing the RESTful approach and JSON payloads for efficient data exchange.

5.2.3 UI and UX

We meticulously design the Front-end to align with key UI and UX principles, ensuring a user-friendly and comprehensible interface.

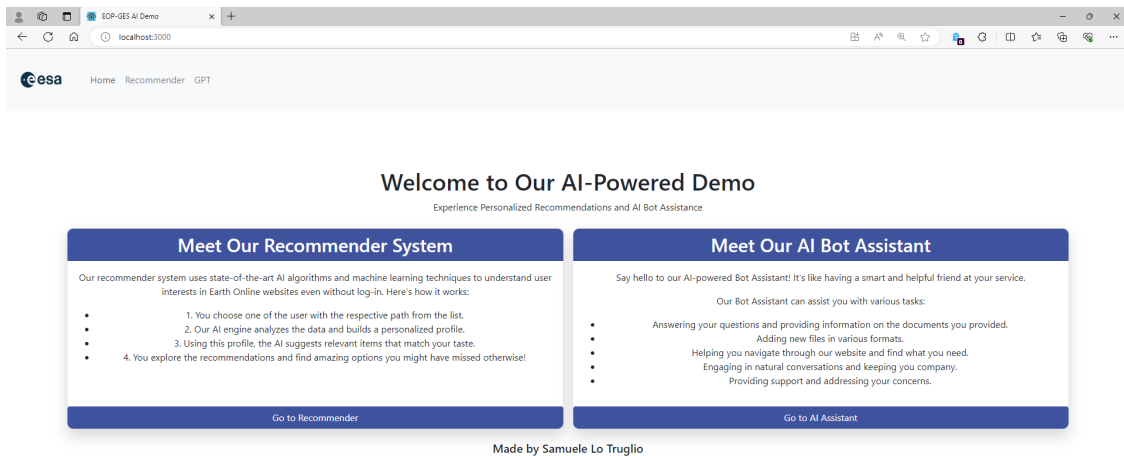


Figure 5.2. Demo Home Page.

As depicted in Figure 5.2, the homepage provides a concise introduction

to the two projects, elucidating their objectives, anticipated outcomes, and the steps required to engage with them. Direct links to the project pages are conveniently placed in the header's navigation bar, allowing users to effortlessly navigate between web pages. Additionally, links to the project pages are thoughtfully positioned at the conclusion of each project description, creating a coherent user journey for readers.

EO Recommender

The challenge arises when attempting to simulate the output of a recommendation system not yet in production. Typically, website recommendations are generated by intricate algorithms that rely on user cookies or logged-in user data. However, these conventional methods are not applicable in this case. To address this, we implement a workaround to mimic user behavior by selecting a visitor.

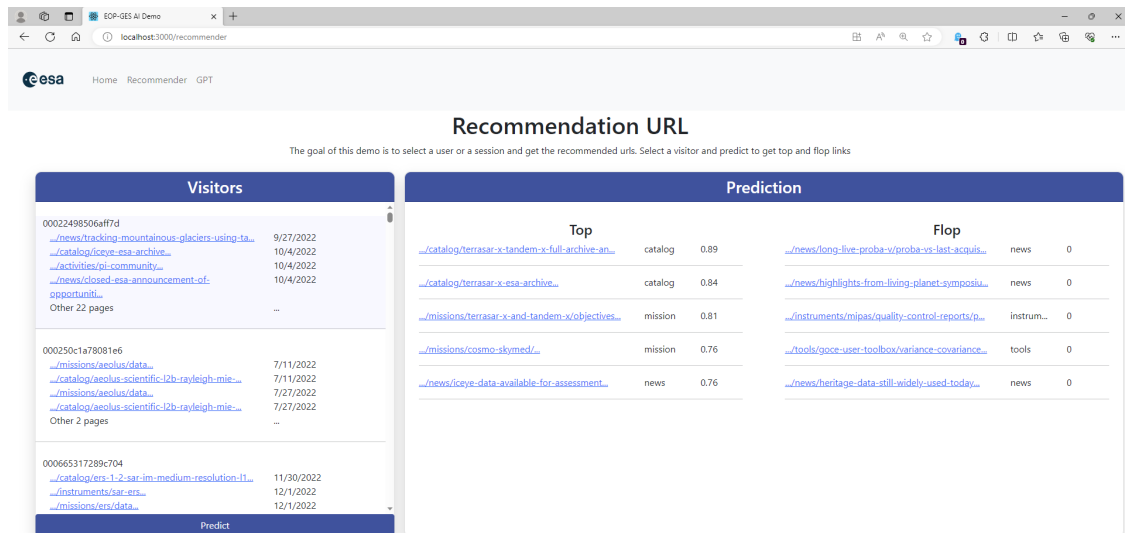


Figure 5.3. Recommender Demo

The Recommender page is organized into two distinct sections: "Visitors" and "Prediction," as illustrated in Figure 5.3. Each section serves a specific purpose. In the "Visitors" section, a list of visitors is displayed, with each visitor identified by a unique visitor ID. This list includes details about the visitor's history of interactions with Earth Online and the related visit date.

Given the potential size of this list, users can scroll through it and select a preferred visitor. Once a visitor is selected, the "Predict" button becomes active, allowing users to trigger the generation of recommendations tailored to that specific visitor.

The resulting predictions are presented in the "Predictions" area of the page. Here, the links are categorized into "top" and "flop," indicating highly recommended and less recommended pages, respectively. Each link displays its associated category and score. Users can click on these links, redirecting them to the corresponding Earth Online pages for further exploration.

EOP-GES Assistant

Visualizing a virtual assistant proves to be a more straightforward challenge compared to the previous one. The concept is to present user questions and model responses in a chat-like interface, resulting in the design shown in Figure 5.4. *For privacy reasons, filenames are obscured in the figure.*

In this demonstration, users have the possibility to view their previously ingested documents on the left side of the interface, with icons denoting the format of the loaded document. If desired, users can also upload new documents, triggering a new ingestion process. Once ingested, the newly added document will appear among the loaded files.

The chat section of the system employs icons to differentiate between user and assistant messages. Messages are displayed within containers with distinct backgrounds and fonts to make it clear who sent each message. Additionally, the typewriter-style font used for the assistant's messages serves to remind users that they are interacting with a machine.

The assistant's replies are presented with the answer, followed by the sources used to generate the answer and the time it took to produce the response. It's important to note that the fragments of sources have been omitted for two reasons: they may contain unusual characters or incorrect formatting due to parsing, and displaying them could result in excessively long output. The inclusion of the response time serves presentation purposes, allowing users to gauge the actual time taken for each answer in case the answer was cached.

Users can input their questions in the chat via the text box at the bottom of the interface and submit them using the right-facing arrow icon, similar to common chat applications. While waiting for a response from the assistant, a series of three dots is displayed to indicate that the system is processing the request, reassuring users that an answer will be provided.

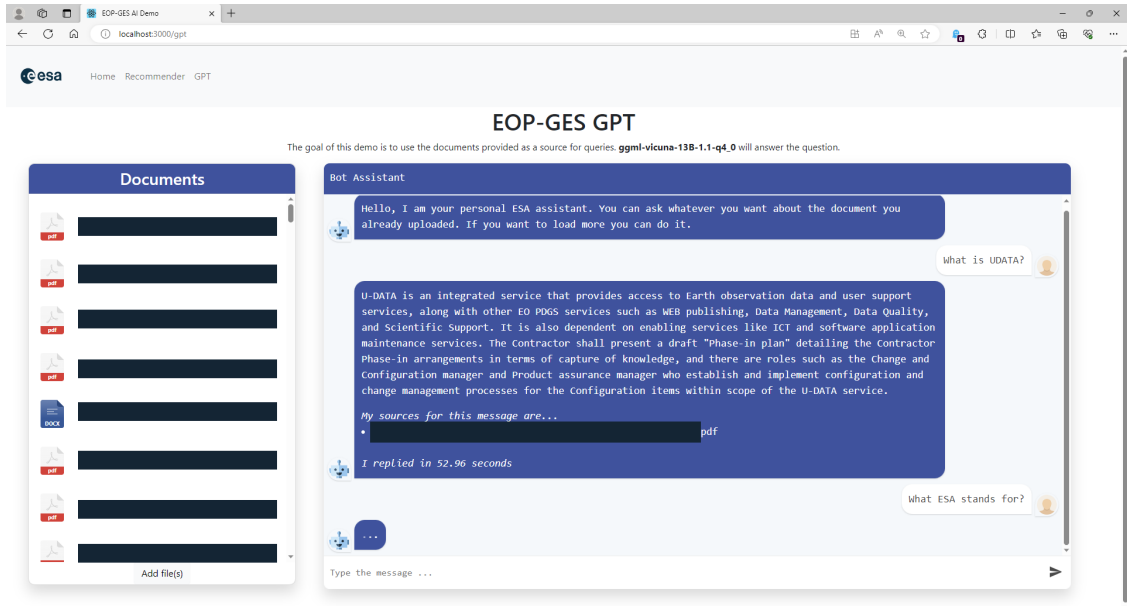


Figure 5.4. Virtual Assistant Demo

5.2.4 Users Feedback

The proposed solution undergoes several phases of presentation and user interaction, starting with demonstrations to potential users during its development and later through screenshots and videos in the final presentation.

In the final presentation, we prepare the audience for the demonstration through a series of slides that include a brief academic introduction to the two types of artificial intelligence models used and a detailed description of each stage of the project’s work, mirroring the structure of this thesis. Then we show the demonstration as the culmination of the presentation.

The users present at the final presentation express strong interest in the projects. They ask questions and propose potential applications, indicating a desire to explore the concepts further. Some users also have the opportunity to test the system, finding it user-friendly and intuitive. They appreciate the interactivity and are not deterred by the waiting time for the virtual assistant’s responses, but rather excited to see the answers.

Our decision to provide a working demonstrator is deemed successful, garnering support from all participants in the presentation. Offering users the opportunity to experience a real AI-based application allows them to grasp the practicality and potential of these technologies beyond mere buzzwords. It serves as a tangible example of how AI can be applied to enhance existing

systems.

However, we observe that the virtual assistant garners greater interest than the recommendation system. This can be attributed to two main factors, which extend beyond the demo itself. First, we don't show the recommendation system within a real application context, as it's showcased in isolation. It's not demonstrated in its intended context, such as within Earth Online's interface, which could provide a more immersive experience. Second, recommendation systems have become commonplace in today's world, with users accustomed to encountering them in popular e-commerce and media streaming services. In contrast, virtual assistants, while gaining popularity, are still relatively novel and less ubiquitous in everyday applications.

Chapter 6

Conclusions

Introducing Artificial Intelligence into new contexts is a common objective for businesses, driven by both the growing interest in AI and the potential value it can bring to existing solutions.

The internship and the resulting work yield valuable insights into the practical applications of AI, considering both its benefits and limitations. The decision to explore applications related to user experience proved to be a wise choice. Within the context of a regulatory environment, the size of the Agency, and the multitude of personnel involved, many initiatives can be challenging to implement or achieve.

One notable challenge is the ever-evolving landscape of technology. The rapid pace of technological advancements often renders previously established solutions obsolete, as newer, more efficient methods become available. This dynamic is particularly evident in the field of AI, which has the potential to revolutionize numerous areas by outperforming previous solutions. Therefore, staying current with AI developments is crucial to harnessing its full potential for the benefit of organizations and users alike.

6.1 EO Recommender

The EO Recommender project faces significant challenges throughout its development, from its inception to the demonstrator stage. Despite these challenges, we underscore the vital role of analytics, highlighting that data collection goes beyond merely tracking the most visited pages. Analytics serve as a valuable source of insight into user behaviors and preferences. Leveraging AI with collected data offers an innovative approach to deriving value from this information.

Utilizing visitors’ navigation history to suggest additional pages for both new and returning users is a critical feature for a website like Earth Online, catering to students, scientists, and enthusiasts. This feature enhances user engagement and recognition of the effort invested in curating and publishing content on the website.

The process of creating the dataset and the methodology employed for model training, tuning, and testing adhered to establish guidelines. The use of updated tools ensures efficiency and lie the foundation for future work to build upon without the need for a complete reevaluation of the process.

SLi-Rec demonstrates impressive performance in a test environment, suggesting its potential for real-world applications. Implementing this model in a live environment could validate the accuracy of suggested pages and yield even more valuable results. SLi-Rec’s advantage likely stems from its time-aware nature, which aligns well with the specific task of recommending pages.

Nevertheless, the research is ongoing, and further improvements can be made. Challenges related to dataset features can be addressed by providing more detailed items, enabling the adoption of alternative models that may yield even better results. Additionally, deeper hyperparameter tuning could lead to improved performance.

A feature that we suggest like a straightforward enhancement for the recommendation system is to incorporate information about what users do not click, as depicted in Figure 6.1. This approach enables the system to penalize pages that are suggested but consistently considers unclicked links by users, further refining the quality of recommendations.

ESA staff responsible for web services and higher-level decision-making recognize the value of recommendations and their potential to enhance the user experience on the website. As a result, they are considering improvements to the current Earth Online system and exploring AI implementations further.

6.2 EOP-GES Assistant

The genesis of this project traces back to our journey at ESA, coupled with the recent advancements in chatGPT and AI technologies. The initial idea meets with interest and curiosity, and the presentation further takes the curiosity of the audience. After the presentation, participants display a heightened interest in the topic, particularly due to the live demonstrator.

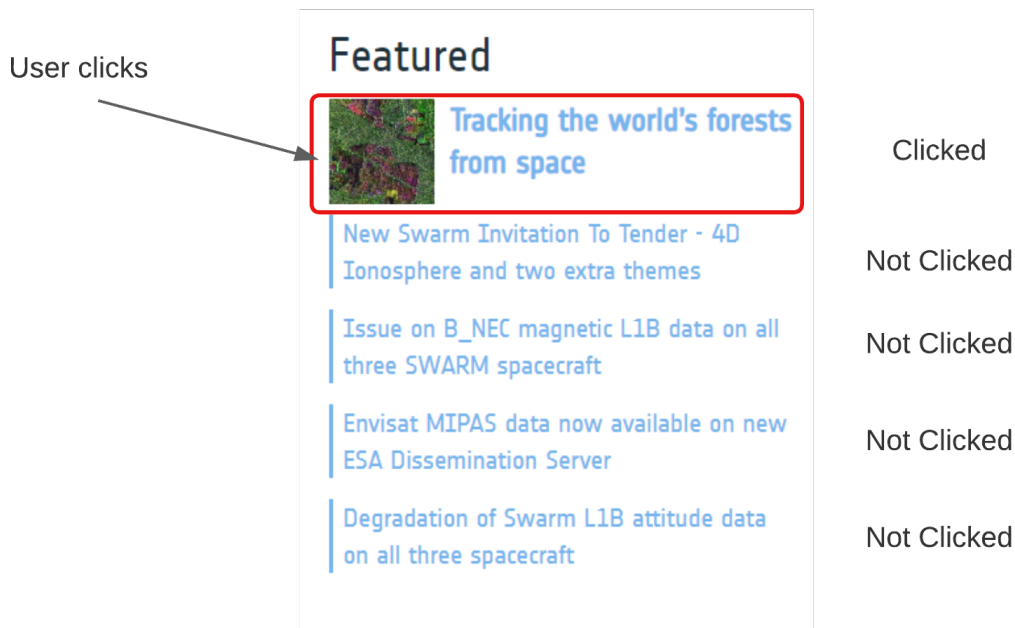


Figure 6.1. Example of information to provide

Large Language Models like GPT hold immense potential in various domains, and the developed project represents just a small glimpse of their possible applications. Despite the initial challenges, including the absence of GPU support, the terminal approach and the default model gpt4all-j-v1.3-groovy with poor performance, the subsequent enhancements and research shed light on the potential of this technology to support users and enhance their work.

Both Vicuna and LLaMA2 demonstrate outstanding performance, and further configuration adjustments could enhance the quality of answers they provide.

However, a critical challenge lies in power consumption, especially when considering scalability. Future development efforts will undoubtedly focus on optimization and reducing the resource demands of the system.

6.3 Potential Developments

During the presentation, we discuss several potential implementations for the two projects, considering existing technologies without the need to address reported issues.

EO Recommender. To enhance the EO Recommender, one possible implementation involves providing recommendations based on user cookies to recognize their browsing history. This would offer users a valuable way to discover more information related to their interests. In the current solution (as exemplified in Figure 6.2), users need to make at least two clicks to access related pages. This not only consumes time but also necessitates loading the search page and the desired page itself. The core usability issue lies in the user having to decide which page to open, as keywords alone cannot be considered recommendations. The absence of such recommendations may lead users to leave the page in search of more relevant content.

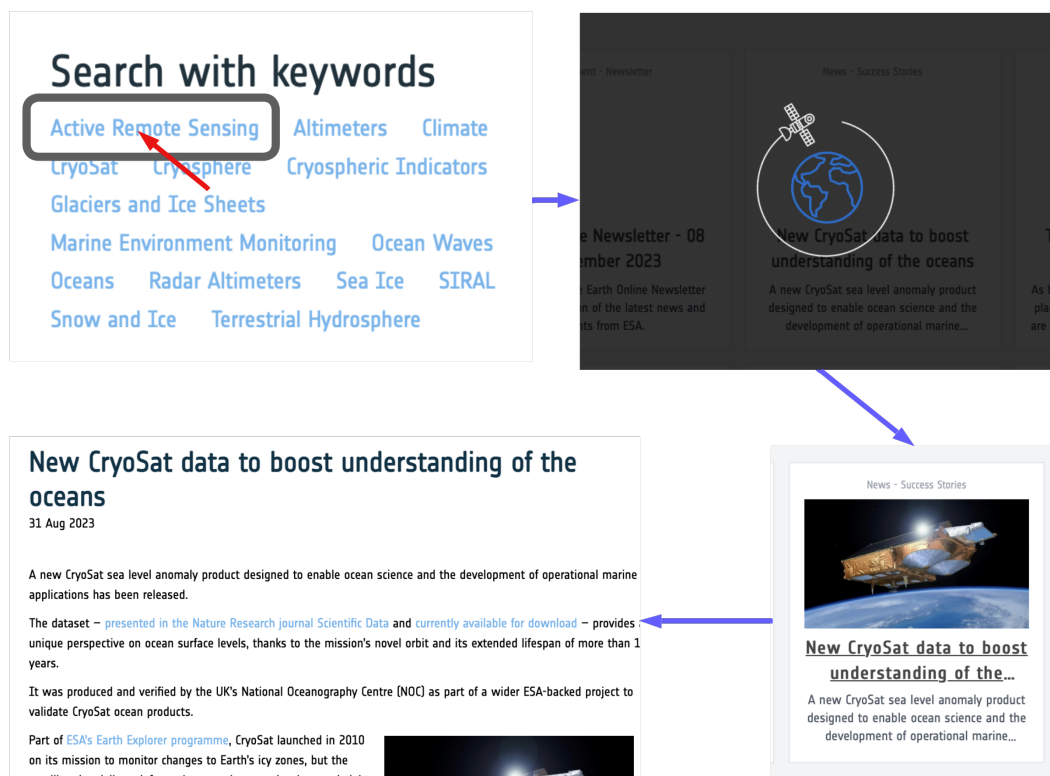


Figure 6.2. Example of navigation to get a related page.

On the contrary, embedding recommendations directly within the web-page content would swiftly offer users related pages, significantly enhancing their experience. Visitors are more likely to click on these recommendations, leading to increased user satisfaction as they easily find what they are looking for, ultimately boosting user engagement.

Virtual Assistant. For the Virtual Assistant, many possible applications can be thought and some examples are:

- **Earth Online Assistant.** Incorporating a chatbot into Earth Online, as initially envisioned (Figure 6.3, could significantly enhance the user experience.

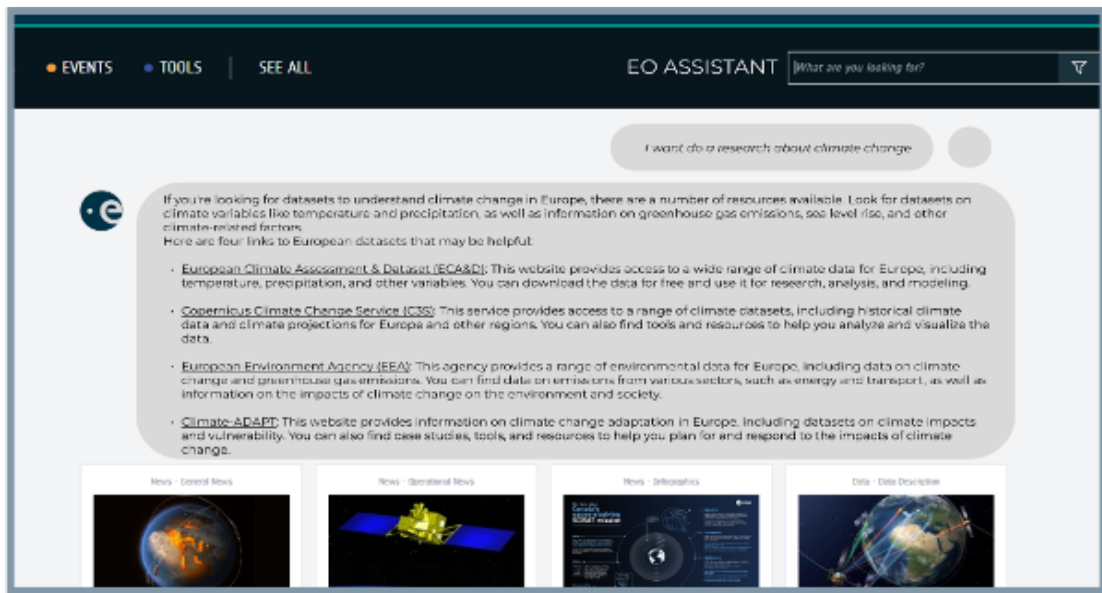


Figure 6.3. First sketch of Earth Online Assistant

By leveraging the knowledge gained from the EOP-GES Assistant project, it's possible to provide to a virtual assistant Earth Online's entire HTML pages repository. This would allow users to interact with a virtual assistant capable of generating summaries and offering previews of the source pages. Users could then choose to open those pages for more information, resulting in improved usability, user engagement, and overall user satisfaction for Earth Online.

- **Shared Knowledge Database.** Many managers at ESA attend meetings where they explain their work and present results. However, managers not directly involved in these activities may lack comprehensive knowledge about contracts and related activities. They often rely on asking questions during meetings or sifting through extensive documents. To address this, it could be created a shared database for these documents and use it for Virtual Assistant ingestion. This would enable other managers to familiarize themselves with the meeting topics in advance, promoting more informed and productive discussions.
- **Newcomers Support.** Welcoming newcomers at ESA can be challenging due to the steep learning curve. Providing a virtual assistant for newcomers would be immensely helpful. They could easily inquire about contracts, acronyms, and other essential information, saving valuable time. Additionally, the assistant could offer the option to read the source documents of its answers, facilitating better understanding. This approach would significantly reduce the learning curve for newcomers, making it easier for them to integrate into the ESA environment.

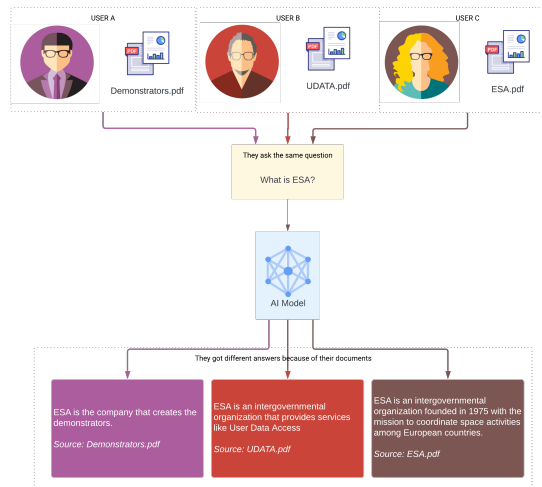


Figure 6.4. Example of individualized assistance.

- **Individualized Assistance.** Implementing a single model that relies on multiple users' document collections is an intriguing idea. Each user could have a private document collection, resulting in personalized answers based on their specific needs. Even if users ask the same question,

they would receive different answers with varying levels of detail, as illustrated in Figure 6.4. This approach could optimize resource utilization compared to deploying separate models for each user type. It would be an excellent solution for enhancing user-specific assistance while minimizing power consumption and resource usage.

In conclusion, the EO Recommender and EOP-GES Assistant projects represent groundbreaking strides in harnessing the power of AI to revolutionize how ESA operates. The EO Recommender not only enhances user experience by providing tailored recommendations but also opens new horizons for understanding user preferences and behaviors. Meanwhile, the Virtual Assistant project stands poised to become a game-changer, offering dynamic knowledge dissemination, meeting optimization, onboarding support, and a truly personalized user experience. Our projects serve as powerful examples of how AI can bring efficiency, intelligence, and innovation to an organization as complex and forward-thinking as ESA. As these technologies continue to evolve, they hold the promise of reshaping how ESA and similar organizations operate, unleashing a future of unprecedented efficiency and productivity.

Bibliography

- [1] Michael J. Pazzani and Daniel Billsus. «Content-Based Recommendation Systems». In: *The Adaptive Web*. 2007. URL: <https://api.semanticscholar.org/CorpusID:7364032>.
- [2] Michael Ekstrand, John Riedl, and Joseph Konstan. «Collaborative Filtering Recommender Systems». In: *Foundations and Trends in Human-Computer Interaction* 4 (Jan. 2011), pp. 175–243. DOI: [10.1561/1100000009](https://doi.org/10.1561/1100000009).
- [3] Ben Schafer et al. «Collaborative Filtering Recommender Systems». In: Jan. 2007.
- [4] Robin Burke. «Hybrid Recommender Systems: Survey and Experiments». In: *User Modeling and User-Adapted Interaction* 12 (Nov. 2002). DOI: [10.1023/A:1021240730564](https://doi.org/10.1023/A:1021240730564).
- [5] Dietmar Jannach, Malte Ludewig, and Lukas Lerche. «Session-based item recommendation in e-commerce: on short-term intents, reminders, trends and discounts». In: *User Modeling and User-Adapted Interaction* 27 (Dec. 2017). DOI: [10.1007/s11257-017-9194-1](https://doi.org/10.1007/s11257-017-9194-1).
- [6] Shoujin Wang et al. «Sequential Recommender Systems: Challenges, Progress and Prospects». In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Aug. 2019. DOI: [10.24963/ijcai.2019/883](https://doi.org/10.24963/ijcai.2019/883). URL: <https://doi.org/10.24963/ijcai.2019/883>.
- [7] Shoujin Wang et al. *A Survey on Session-based Recommender Systems*. 2021. arXiv: [1902.04864 \[cs.IR\]](https://arxiv.org/abs/1902.04864).
- [8] Malte Ludewig et al. «Performance comparison of neural and non-neural approaches to session-based recommendation». In: Sept. 2019, pp. 462–466. ISBN: 978-1-4503-6243-6. DOI: [10.1145/3298689.3347041](https://doi.org/10.1145/3298689.3347041).

- [9] Malte Ludewig et al. «Empirical analysis of session-based recommendation algorithms». In: *User Modeling and User-Adapted Interaction* 31.1 (Oct. 2020), pp. 149–181. DOI: [10.1007/s11257-020-09277-1](https://doi.org/10.1007/s11257-020-09277-1). URL: <https://doi.org/10.1007/s11257-020-09277-1>.
- [10] Zeping Yu et al. «Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation». In: Aug. 2019, pp. 4213–4219. DOI: [10.24963/ijcai.2019/585](https://doi.org/10.24963/ijcai.2019/585).
- [11] Steffen Rendle et al. *BPR: Bayesian Personalized Ranking from Implicit Feedback*. 2012. arXiv: [1205.2618](https://arxiv.org/abs/1205.2618) [cs.IR].
- [12] Jiayi Tang and Ke Wang. *Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding*. 2018. arXiv: [1809.07426](https://arxiv.org/abs/1809.07426) [cs.IR].
- [13] Balázs Hidasi et al. *Session-based Recommendations with Recurrent Neural Networks*. 2016. arXiv: [1511.06939](https://arxiv.org/abs/1511.06939) [cs.LG].
- [14] Fajie Yuan et al. *A Simple Convolutional Generative Network for Next Item Recommendation*. 2018. arXiv: [1808.05163](https://arxiv.org/abs/1808.05163) [cs.IR].
- [15] Jianxun Lian et al. *Multi-Interest-Aware User Modeling for Large-Scale Sequential Recommendations*. 2021. arXiv: [2102.09211](https://arxiv.org/abs/2102.09211) [cs.IR].
- [16] Hongwei Wang et al. *DKN: Deep Knowledge-Aware Network for News Recommendation*. 2018. arXiv: [1801.08284](https://arxiv.org/abs/1801.08284) [stat.ML].
- [17] Guolin Ke et al. «LightGBM: A Highly Efficient Gradient Boosting Decision Tree». In: *NIPS*. 2017. URL: <https://api.semanticscholar.org/CorpusID:3815895>.
- [18] Chuhan Wu et al. *Neural News Recommendation with Attentive Multi-View Learning*. 2019. arXiv: [1907.05576](https://arxiv.org/abs/1907.05576) [cs.CL].
- [19] Steffen Rendle. «Factorization Machines». In: Dec. 2010, pp. 995–1000. DOI: [10.1109/ICDM.2010.127](https://doi.org/10.1109/ICDM.2010.127).
- [20] Maciej Kula. *Metadata Embeddings for User and Item Cold-start Recommendations*. 2015. arXiv: [1507.08439](https://arxiv.org/abs/1507.08439) [cs.IR].
- [21] Heng-Tze Cheng et al. *Wide & Deep Learning for Recommender Systems*. 2016. arXiv: [1606.07792](https://arxiv.org/abs/1606.07792) [cs.LG].
- [22] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: [2005.14165](https://arxiv.org/abs/2005.14165) [cs.CL].
- [23] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: [2303.08774](https://arxiv.org/abs/2303.08774) [cs.CL].

BIBLIOGRAPHY

- [24] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: [2302.13971](https://arxiv.org/abs/2302.13971) [cs.CL].
- [25] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: [2307.09288](https://arxiv.org/abs/2307.09288) [cs.CL].
- [26] Yunxiang Li et al. *ChatDoctor: A Medical Chat Model Fine-Tuned on a Large Language Model Meta-AI (LLaMA) Using Medical Domain Knowledge*. 2023. arXiv: [2303.14070](https://arxiv.org/abs/2303.14070) [cs.CL].
- [27] Sida Peng et al. *The Impact of AI on Developer Productivity: Evidence from GitHub Copilot*. 2023. arXiv: [2302.06590](https://arxiv.org/abs/2302.06590) [cs.SE].
- [28] A. Shaji George, A. S. Hovan George, and A. S. Gabrio Martin. «The Environmental Impact of AI: A Case Study of Water Consumption by Chat GPT». In: *Partners Universal International Innovation Journal* 1.2 (Apr. 2023), pp. 97–104. DOI: [10.5281/zenodo.7855594](https://doi.org/10.5281/zenodo.7855594). URL: <https://puuij.com/index.php/research/article/view/39>.
- [29] David Glukhov et al. *LLM Censorship: A Machine Learning Challenge or a Computer Security Problem?* 2023. arXiv: [2307.10719](https://arxiv.org/abs/2307.10719) [cs.AI].
- [30] Lixiang Yan et al. *Practical and Ethical Challenges of Large Language Models in Education: A Systematic Scoping Review*. 2023. arXiv: [2303.13379](https://arxiv.org/abs/2303.13379) [cs.CL].
- [31] Laura Weidinger et al. *Ethical and social risks of harm from Language Models*. 2021. arXiv: [2112.04359](https://arxiv.org/abs/2112.04359) [cs.CL].
- [32] *ChatGPT*. Openai.com, 2023. URL: <https://chat.openai.com/> (visited on 09/18/2023).
- [33] *Bard*. Google.com, 2023. URL: <https://bard.google.com/?hl=it> (visited on 09/18/2023).
- [34] Ruiyun Xu, Yue Feng, and Hailiang Chen. *ChatGPT vs. Google: A Comparative Study of Search Performance and User Experience*. 2023. arXiv: [2307.01135](https://arxiv.org/abs/2307.01135) [cs.AI].
- [35] elasticsearch. *elasticsearch/elasticsearch*. 2015. URL: <https://github.com/elasticsearch/elasticsearch>.
- [36] Iván Martínez and other contributors. *privateGPT*. <https://github.com/imartinez/privateGPT>. 2023.
- [37] chroma-core. *chroma*. <https://github.com/chroma-core/chroma>. 2022.

- [38] Shoujin Wang et al. «Sequential Recommender Systems: Challenges, Progress and Prospects». In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Aug. 2019. DOI: [10.24963/ijcai.2019/883](https://doi.org/10.24963/ijcai.2019/883). URL: <https://doi.org/10.24963/2Fijcai.2019%2F883>.
- [39] Rui He and Julian McAuley. «Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering». In: *Proceedings of the 25th International Conference on World Wide Web*. ACM, 2016.
- [40] Julian McAuley et al. «Image-based recommendations on styles and substitutes». In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015.
- [41] Wenjing Meng, Deqing Yang, and Yanghua Xiao. «Incorporating User Micro-behaviors and Item Knowledge into Multi-task Learning for Session-based Recommendation». In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, July 2020. DOI: [10.1145/3397271.3401098](https://doi.org/10.1145/3397271.3401098). URL: <https://doi.org/10.1145%2F3397271.3401098>.
- [42] Scott Graham, Jun-Ki Min, and Tao Wu. «Microsoft Recommenders: Tools to Accelerate Developing Recommender Systems». In: *Proceedings of the 13th ACM Conference on Recommender Systems*. RecSys '19. Copenhagen, Denmark: Association for Computing Machinery, 2019, pp. 542–543. ISBN: 9781450362436. DOI: [10.1145/3298689.3346967](https://doi.org/10.1145/3298689.3346967). URL: <https://doi.org/10.1145/3298689.3346967>.
- [43] Wang-Cheng Kang and Julian McAuley. *Self-Attentive Sequential Recommendation*. 2018. arXiv: [1808.09781](https://arxiv.org/abs/1808.09781) [cs.IR].
- [44] Liwei Wu et al. «SSE-PT: Sequential Recommendation Via Personalized Transformer». In: Sept. 2020, pp. 328–337. DOI: [10.1145/3383313.3412258](https://doi.org/10.1145/3383313.3412258).
- [45] Tom Fawcett. «An introduction to ROC analysis». In: *Pattern Recognition Letters* 27.8 (2006). ROC Analysis in Pattern Recognition, pp. 861–874. ISSN: 0167-8655. DOI: <https://doi.org/10.1016/j.patrec.2005.10.010>. URL: <https://www.sciencedirect.com/science/article/pii/S016786550500303X>.

- [46] Shuheï Watanabe. *Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance*. 2023. arXiv: [2304.11127](https://arxiv.org/abs/2304.11127) [cs.LG].
- [47] Microsoft. *Neural Network Intelligence*. Version 2.0. Jan. 2021. URL: <https://github.com/microsoft/nni>.
- [48] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: [1908.10084](https://arxiv.org/abs/1908.10084) [cs.CL].
- [49] Yuvanesh Anand et al. *GPT4All: Training an Assistant-style Chatbot with Large Scale Data Distillation from GPT-3.5-Turbo*. <https://github.com/nomic-ai/gpt4all>. 2023.
- [50] Ben Wang. *Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX*. <https://github.com/kingoflolz/mesh-transformer-jax>. May 2021.
- [51] Thomas Wolf et al. *HuggingFace’s Transformers: State-of-the-art Natural Language Processing*. 2020. arXiv: [1910.03771](https://arxiv.org/abs/1910.03771) [cs.CL].
- [52] *Hugging face – the AI community building the future*. <https://huggingface.co/>. Accessed: 2023-09-07.
- [53] Wei-Lin Chiang et al. *Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality*. Mar. 2023. URL: <https://lmsys.org/blog/2023-03-30-vicuna/>.
- [54] Subhabrata Mukherjee et al. *Orca: Progressive Learning from Complex Explanation Traces of GPT-4*. 2023. arXiv: [2306.02707](https://arxiv.org/abs/2306.02707) [cs.CL].
- [55] Sam Witteveen and Martin Andrews. *Investigating Prompt Engineering in Diffusion Models*. 2022. arXiv: [2211.15462](https://arxiv.org/abs/2211.15462) [cs.CV].
- [56] Ashraf Elnashar et al. «Question Formulation and Transformer Model Resilience: Effects of Question-Wording Variants on the Robustness of Transformer-Based Question-Answer Models». In: *2022 International Conference on Computational Science and Computational Intelligence (CSCI)* (2022), pp. 1796–1799. URL: <https://api.semanticscholar.org/CorpusID:253380825>.
- [57] Edward Y. Chang. *Prompting Large Language Models With the Socratic Method*. 2023. arXiv: [2303.08769](https://arxiv.org/abs/2303.08769) [cs.LG].
- [58] F. Pedregosa et al. «Scikit-learn: Machine Learning in Python». In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

BIBLIOGRAPHY

- [59] *Summarization*. https://python.langchain.com/docs/use_cases/summarization. Accessed: 2023-09-07.
- [60] *Prompt Template*. https://python.langchain.com/docs/modules/model_io/prompts/prompt_templates/. Accessed: 2023-09-07.
- [61] Dirk Merkel. «Docker: lightweight linux containers for consistent development and deployment». In: *Linux journal* 2014.239 (2014), p. 2.
- [62] Facebook. *React*. Version 18.2.0. URL: [%5Curl%7Bhttps://github.com/facebook/react%7D](https://github.com/facebook/react).
- [63] *Bootstrap*. <https://getbootstrap.com/>. Accessed: 2023-09-07.
- [64] *Express.js*. <https://expressjs.com/>. Accessed: 2023-09-07.
- [65] *MongoDB*. <https://www.mongodb.com/>. Accessed: 2023-09-07.
- [66] Pallets. *Flask*. Version 2.2.5. URL: [%5Curl%7Bhttps://github.com/pallets/flask/%7D](https://github.com/pallets/flask/).
- [67] Rui Pereira et al. «Energy Efficiency across Programming Languages: How Do Energy, Time, and Memory Relate?» In: *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*. SLE 2017. Vancouver, BC, Canada: Association for Computing Machinery, 2017, pp. 256–267. ISBN: 9781450355254. DOI: [10.1145/3136014.3136031](https://doi.org/10.1145/3136014.3136031). URL: <https://doi.org/10.1145/3136014.3136031>.

Appendix A

Appendix

In the following tables and images are shown further work that completes the other showed in the thesis.

A.1 Virtual Assistant

Figure [A.1](#) shows answer length, answer similarity and time grouped by model in different ways. So the results are averaged considering all the questions.

Figure [A.2](#) shows answer length, similarity and time grouped by the original question, so results are averaged across all the models.

Starting from `tab:questions-1` there are all the original questions on the left column and the augmented versions on the right.

Table [A.1](#) shows the prompt templates used for the models.

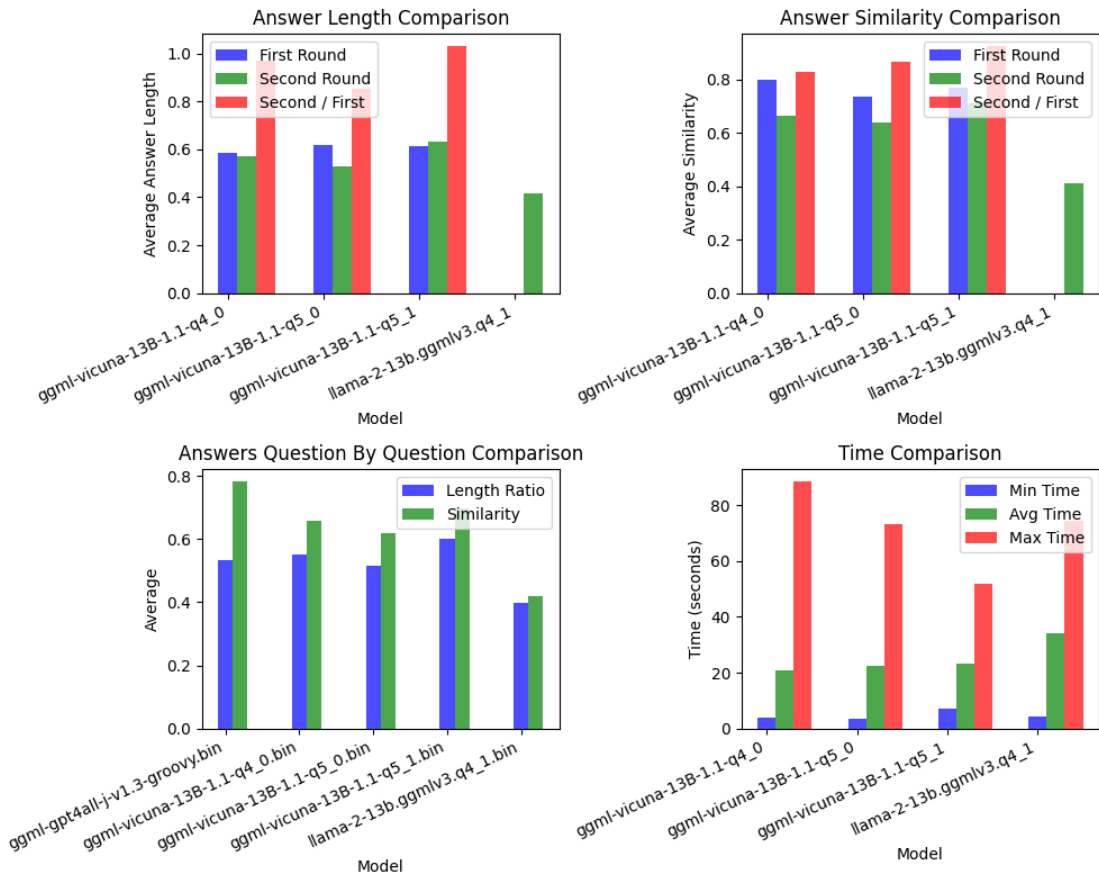


Figure A.1. Assistant Testing grouped by model

A.1 – Virtual Assistant

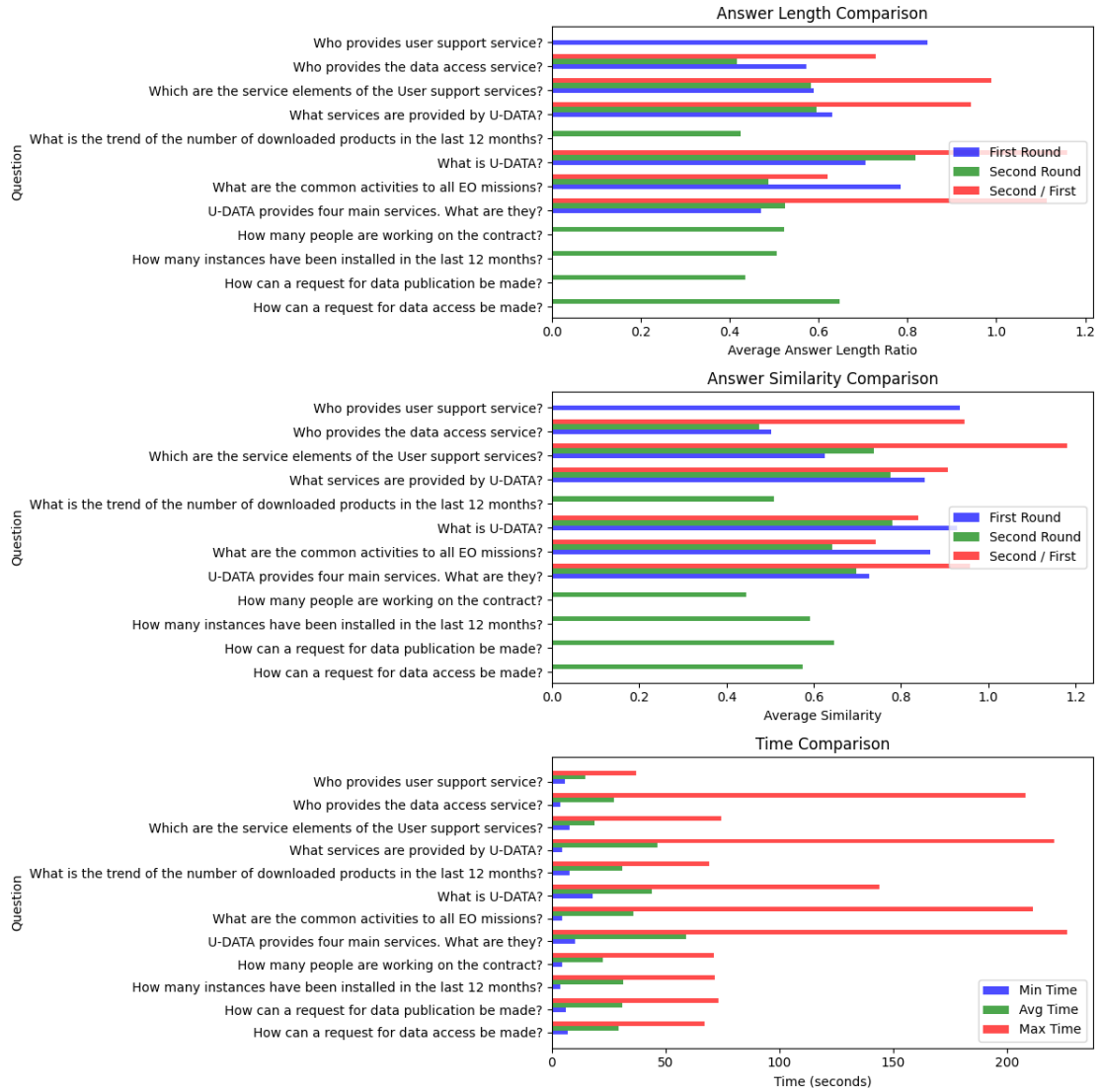


Figure A.2. Assistant Testing grouped by question

Question	Augmented
What is U-DATA?	What is U-DATA? What is UDATA?
Who provides user support service?	Who provides user support service?
What services are provided by U-DATA?	What services are provided by U-DATA? What are the services offered by U-DATA? What does U-DATA provide in terms of services? What types of services does U-DATA offer? What service offerings are available from U-DATA?
U-DATA provides four main services. What are they?	U-DATA provides four main services. What are they? What are the four main services provided by U-DATA? What are the primary services offered by U-DATA? What are the main service offerings of U-DATA? U-DATA offers four key services. What are they?
Who provides the data access service?	Who provides the data access service? Who is responsible for providing the data access service? Which entity offers the data access service? Who offers the service of data access? Who provides the service for accessing data?

Table A.1. Original Questions and Augmented ones. Part 1/4

Question	Augmented
What are the common activities to all EO missions?	<p>What are the common activities to all EO missions?</p> <p>What activities are common to all EO missions?</p> <p>What are the shared activities across all EO missions?</p> <p>What activities are typically found in every EO mission?</p> <p>What are the commonly performed activities in EO missions?</p>
Which are the service elements of the User support services?	<p>Which are the service elements of the User support services?</p> <p>What are the service elements included in the User support services?</p> <p>Which specific service elements are part of the User support services?</p> <p>What are the constituent components of the User support services?</p> <p>Which service elements are encompassed within the User support services?</p>
What is the trend of the number of downloaded products in the last 12 months?	<p>What is the trend of the number of downloaded products in the last 12 months?</p> <p>What has been the pattern of product downloads over the past year?</p> <p>How has the number of downloaded products changed in the last 12 months?</p> <p>Can you describe the variation in the number of products downloaded over the past year?</p> <p>What has been the overall impact of the product downloads in the past year?</p>

Table A.2. Original Questions and Augmented ones. Part 2/4

Question	Augmented
How many instances have been installed in the last 12 months?	<p>How many instances have been installed in the last 12 months?</p> <p>In the last 12 months, how many instances have been installed?</p> <p>How many installations of instances have occurred in the past year?</p> <p>Can you provide information on the number of instances installed in the last 12 months?</p> <p>How many times have instances been installed in the past year?</p>
How many people are working on the contract?	<p>How many people are working on the contract?</p> <p>How many employees are currently assigned to the contract?</p> <p>Could you tell me the current workforce dedicated to the contract?</p> <p>What is the current count of personnel assigned to the contract?</p> <p>How large is the workforce currently engaged in the contract?</p>

Table A.3. Original Questions and Augmented ones. Part 3/4

Question	Augmented
How can a request for data publication be made?	How can a request for data publication be made? What is the process for submitting a data publication request? Can you explain the steps to request data publication? How can one go about making a data publication request? What is the procedure for submitting a data publication request?
How can a request for data access be made?	How can a request for data access be made? How can one request access to the data? What are the steps to request data access? Is there a procedure for seeking access to the data? How does one initiate the process of requesting data access?

Table A.4. Original Questions and Augmented ones. Part 4/4

Model	Prompt Template
ggml-vicuna-13B-1.1-q X _ X	<i>Not Needed</i>
llama-2-13b.ggmlv3.q X _1	<i>Not Needed</i>
llama-2-13b-chat.ggmlv3.q4_1	SYSTEM: You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature. If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information. Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer. <i>context</i> USER: <i>question</i> ASSISTANT:
openassistant-llama2-13b-orca-8k-3319.ggmlv3.q4_1	You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature. If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information. Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer. <i>context</i> </s> <i>question</i> </s>

Table A.5. Employed prompt templates