Chatbot Sophia: A New Perspective on Providing User Control in Recommendation Systems for E-Commerce

Yuchuan Lin
Trinity University, stanleylin28@gmail.com

Follow this and additional works at: https://digitalcommons.trinity.edu/compsci_honors

Recommended Citation
https://digitalcommons.trinity.edu/compsci_honors/60

This Thesis open access is brought to you for free and open access by the Computer Science Department at Digital Commons @ Trinity. It has been accepted for inclusion in Computer Science Honors Theses by an authorized administrator of Digital Commons @ Trinity. For more information, please contact jcostanz@trinity.edu.
Chatbot Sophia: A New Perspective on Providing User Control in Recommendation Systems for E-Commerce

Yuchuan Lin

A DEPARTMENT HONORS THESIS SUBMITTED TO THE DEPARTMENT OF Computer Science AT TRINITY UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR GRADUATION WITH DEPARTMENTAL HONORS

DATE __________ 5/12/2021 __________

_______ Dr. Sheng Tan_______ _______ Dr. Yu Zhang_______
THESIS ADVISOR DEPARTMENT CHAIR

____________________________________
Michael Soto, AVPAA
Student Agreement

I grant Trinity University ("Institution"), my academic department ("Department"), and the Texas Digital Library ("TDL") the non-exclusive rights to copy, display, perform, distribute and publish the content I submit to this repository (hereafter called "Work") and to make the Work available in any format in perpetuity as part of a TDL, digital preservation program, Institution or Department repository communication, distribution or preservation effort.

I understand that once the Work is submitted, a bibliographic citation to the Work can remain visible in perpetuity, even if the Work is updated or removed.

I understand that the Work’s copyright owner(s) will continue to own copyright outside these non-exclusive granted rights.

I warrant that:

1) I am the copyright owner of the Work, or  
2) I am one of the copyright owners and have permission from the other owners to submit the Work, or  
3) My Institution or Department is the copyright owner and I have permission to submit the Work, or  
4) Another party is the copyright owner and I have permission to submit the Work.

Based on this, I further warrant to my knowledge:

1) The Work does not infringe any copyright, patent, or trade secrets of any third party,  
2) The Work does not contain any libelous matter, nor invade the privacy of any person or third party, and  
3) That no right in the Work has been sold, mortgaged, or otherwise disposed of, and is free from all claims.

I agree to hold TDL, Institution, Department, and their agents harmless for any liability arising from any breach of the above warranties or any claim of intellectual property infringement arising from the exercise of these non-exclusive granted rights.

I choose the following option for sharing my thesis (required):

[ Y ] Open Access (full-text discoverable via search engines)  
[ ] Restricted to campus viewing only (allow access only on the Trinity University campus via digitalcommons.trinity.edu)

I choose to append the following Creative Commons license (optional): No.
Acknowledgement

I would like to express my deepest gratitude to my thesis advisor, Dr. Tan, for his patient guidance, enthusiastic encouragement and helpful suggestions of this research work.

I would like to extend my thanks to Dr. Zhang and Dr. Lewis, as you are one of the most important advisors in my life who has been teaching me things way beyond Computer Science over the last few years.

I would also like to express my special thanks of gratitude to Dr. Michael Soto and Ms. Martha DeLeon for providing me an opportunity to finish this work.
Abstract

Nowadays, the visible success of online platforms certainly put a spotlight on the importance of recommendation systems (RS) in multiple domains. Typically, the use of RS has been proved to lead to considerable improvements for e-commerce business by bringing in various types of positive effects. However, recommendation systems are also known to be controversial because of the concerns they introduce: lack of transparency, reduction of diversity, little to no user control, etc. On the other hand, with the significant progress in Natural Language Processing and gradual acceptance of Artificial Intelligence by end-users, one cannot ignore the fact that conversational systems, especially, virtual personal assistants, are drawing more attention in many industries.

In order to alleviate the issues introduced by RS in return for a better user-friendly experience, many researchers are seeking to combine recommendation systems with conversational systems through different means. Aiming towards that same goal, we designed and implemented a simple chatbot, Sophia, for demonstrating the potential of chatbots in improving the user experience for e-commerce platforms in terms of user control. In particular, this work, serving as the groundwork for a series of the proposed research, will focus on the design, current progress, and future plan of both the chatbot and an associated e-commerce website. These two components, along with a simple product recommendation system, were built and integrated altogether into one project: E-Commerce with Sophia (EWS). Aside from achieving user control for e-commerce’s RS via a creative conversation-based
approach, unexpectedly, we discovered that EWS might have the prospect of becoming a general solution to implementing, presenting, and comparing different user control approaches in RS for e-commerce.

**Keywords:** E-Commerce; Chatbot; Recommendation Systems; Conversational Systems; Web Development; Human-Computer interaction; Collaborative Filtering
1 Introduction

With an unabated annual growth that is likely to continue for the coming decades, e-commerce, also known as electronic commerce or internet commerce, has become one of the most successful online businesses globally (Goetsch, 2014). Besides other inherent advantages of e-commerce contributing to this growth, a wide variety of products play a primary role, especially when compared to a physical retail store (Goetsch, 2014). Taking Amazon, the e-commerce giant, as an example, in 2020, the number of products that are available for Amazon customers to purchase is around 12 million across all its main categories and services, and 350 million when including the third-party sellers as well (Dunne, 2021). Nevertheless, when a customer trying to find the desired item, this abundant amount of products suddenly turn into a double-edged sword given the fact that it is extremely challenging and time-consuming to browse through the product list, even with the products being categorized accordingly (Gupta et al., 2015). This phenomenon, normally referred to as information overload, has become one of the most common and important issues in today’s information society (Boehmer et al., 2015).

Fig.1. Products recommended by Amazon’s recommendation system based on browsing history.

In order to deal with information overload, a relatively new technology called recommendation system has been introduced. Typically, recommendation systems are
used to retrieve and rank potential items of interest for each individual user or a set of similar users from all items based on various kinds of information that is either explicit or implicit. Considering the case of Amazon again, by taking the browsing history, purchase records, reviews, inferred user profile, and many other factors into account, the recommendation system of Amazon has already been helping its customers to locate the potential items of interest for almost 20 years (Linden et al., 2003), as shown in Figure 1, contributing about 35 percent of consumer purchases on Amazon (MacKenzie et al., 2013). Moreover, it is worth knowing that a well-designed and fine-tuned recommendation system does not merely bring the increase in sales and revenue to the table. Other explicit positive effects include higher click-through rates, adoption and conversion rates, and better sales distributions (Jannach & Jugovac, 2019). There are implicit positive results as well consisting of increase of trust, acceptance, satisfaction, efficiency, engagement, positive usefulness, better task performance, usability, and more usability feedback for the service provider. (He et al., 2016)

Nevertheless, many researchers have become more aware of the fact that, for obtaining those desired results listed above, i.e., reaching the maximum effectiveness of recommendation systems, some factors other than recommendation accuracy play important roles. In general, these factors, or, challenges, unlike accuracy that can be improved distinctly by developing or deploying better algorithms, are due to the lack of industry standards and attention. In order to enhance RS thoroughly, as accuracy being intensively focused on in the past, these challenges of recommendation systems need to be addressed as well. Among them, the most crucial ones are: (1) transparency and
justification, (2) user control, (3) lack of diversity, (4) cold start issues and (5) contextual information acquisition and representation (He et al., 2016). Notice that many of these challenges, such as user control, are human-related factors since the process of item recommendation inevitably needs to involve human-computer interaction and data visualization. Thereby, lots of works have already been carried out based on this essence. For instance, many researchers are seeking to combine interactive visualization techniques with RS to improve the transparency and controllability of the recommendation process (He et al., 2016). Similarly, the logic also applies to conversational systems.

Conversational systems, or, dialogue systems, are computer systems designed to serve the user through conversation for various purposes given the context. In recent years, being boosted by the significant progress in Natural Language Processing, Voice Recognition, Data Mining, Deep Learning, and cloud computing, intelligent virtual assistants (IVA), like Apple Siri and Amazon Alexa, the most advanced application of conversational systems, have already entered our daily life (Sun & Zhang, 2018). One might even say that IVA are the Artificial Intelligence technology that users familiar with the most nowadays. Furthermore, just as the human conversation is a multimodal interaction, the modes for communication on both the input and output of a conversation system do not merely limit to text and speech. The potential of such a system might just be far beyond all expectations. Therefore, it is foreseeable that, with further development and attention, conversational systems will become increasingly prevalent, versatile, more
accepted by end-users, and even influence how applications in all domains being designed.

In this paper, with the aim of providing e-commerce customers with control over their personalized recommendation strategy, we try to integrate conversational systems, recommendation systems, and e-commerce seamlessly. Considering the contribution of this idea, it is innovative as there is no previous work that seeks to tackle the user control problem in e-commerce's recommendation system through a conversational approach. In particular, we modified an open-source e-commerce website, fed it with actual Amazon product data, added in the product recommendation functionality and embedded our own chatbot called Sophia, which enables users to view or alter their recommendation configuration through simple conversation. After reaching our main goal, we compared the integrated system, E-Commerce with Sophia (EWS), with the works done by others, and realized EWS, due to its design and infrastructure, stands out in some unique aspects, especially the extensibility.

2 Related work

In order to prepare for our work, we deliberately selected and studied several cutting-edge literatures and applications that either being significant to the related fields or giving us research directions. Some of them that contribute to this project the most will be briefly discussed in this section based on the following concepts: user control for recommendation systems, and conversational systems in general.
2.1 User control for recommendation systems

When talking about products or platforms that have already put user control for recommendation systems in production, YouTube is certainly a prominent example. As shown in Figure 2, being provided with a succinct user interface, viewers are able to provide their negative user feedback concerning the recommended videos. These simple user control options, along with “like” and “dislike” buttons, watch history, reviews, and etc., create a comprehensive feedback loop that leads to a more effective RS (Davidson et al., 2010). Similarly, for better understanding the taste of users, the most representative website-based movie recommendation system, MovieLens allows its users to give negative feedback to the recommended movies by clicking a button (Harper & Konstan, 2015). In online shopping, e-commerce websites like Amazon also offer its customers ways to improve recommendations. For example, as shown in Figure 3, there is a page on
Amazon where customers can rate or choose which products they purchased shall not be used for recommendations.

Fig.3. User control over purchased products on Amazon.

Besides its real-world applications, user control for recommendation systems is also a research topic that attracts a lot of attention. A group of researchers conducted an innovative experiment by allowing users to switch the recommendation algorithms of a movie recommendation service freely, as shown in Figure 4 (Ekstrand et al., 2015). They discovered that a significant fraction (72.1%) of users did try out the recommender-switching feature and lots of them expressed interest in knowing what algorithms were used (Ekstrand et al., 2015). Another group of researchers took that even further by adding not only the recommendation algorithms switcher but also a couple of other RS-related filters into a news website prototype as pop-ups, as shown in Figure 5 (Harambam et al., 2019). According to their study, an intelligible user profile, which consists of reading history and flexible preferences setting, and the functionality of recommendation algorithms switching are highly valued by the users.
2.2 Conversational systems

According to Sun, conversational systems can be mainly divided into three categories: chit-chat, informational chat, and task-oriented chat (Sun & Zhang, 2018). All three types of conversational systems can be easily found in various kinds of
contemporary applications and cover their own associated research topics. We focus on finding works related to the last two of them as Chit-chat, aiming to provide informal conversation with users, is none of our concern at this point.

The most well-known examples for informational chat are the intelligent virtual assistants offered by many technology companies, such as Apple Siri. These assistants are able to help the user find information or answer questions according to a simple conversation. Another example of informational chat that is relatively not intelligent but surprisingly helpful is the chatbot embedded in the official website of MongoDB database. As shown in Figure 6, the chatbot is able to provide direct guidance interactively to the user based on the input option he made. For areas like software development where a novice normally finds it is difficult to locate the proper information or content, this sort of support offered by the official is undoubtedly valuable.

![Fig.6. Conversation with MongoDB’s chatbot.](image-url)
Whereas informational chat is getting more common to all kinds of users, task-oriented chat, a kind of conversational systems designed for helping the user fulfill a specific task, is also gaining visibility in multiple domains. In e-commerce setting, a chatbot service that can take orders from the user directly through conversation has been designed and developed (Asadi & Hemadi, 2018). Moreover, back in 2016, clothing retail company, H&M, released its mobile chatbot, which works like a personal stylist who is able to recommend outfits to the user after he answers few questions, as shown in Figure 7 (O’Neill, 2017). Similarly, there is an e-commerce website based chatbot that is capable of picking suitable products for users based on conversation (Gupta et al., 2015). This type of chatbot, which combines recommendation techniques with conversational systems is also referred to as conversational recommendation systems (Jannach et al., 2020).

![Image of H&M chatbot](image)

Fig.7. The H&M chatbot that recommends outfits (O’Neill, 2017).
3 Concept description

3.1 Overview

Despite there are already a few researchers realized the potential of combining conversational techniques with recommendation systems, works seldom focus on how to make the existing way of product recommendation better, which is actually part of the online shopping experience that most e-commerce customers have already accepted and used to for years. We infer that part of the reason may due to the availability of a public e-commerce platform that can be used for conducting related research. Consequently, given the insights from the mentioned literatures and applications, we propose a framework that is capable of enhancing the existing online shopping experience of an e-commerce mock-up website through a simple chatbot, Sophia. Sophia aims to provide e-commerce customers with an innovative functionality: controllability for their personalized recommendation configuration and strategy. The following paragraphs discuss the major components in our framework.

3.2 Components

As shown in Figure 8, our framework consists of mainly three parts: an e-commerce website, a recommendation system, and a chatbot that is embedded into the website.
3.2.1 E-Commerce Website

This component is the mock-up of today’s e-commerce platform and serving as the main interface that users interact with. In detail, it supports the most fundamental functionalities that a typical e-commerce website offers, including product display, product searching, shopping cart, order placement, product management, user management, and product recommendation. All the concepts of this component shall be familiar to anyone who has experience with modern online shopping services. Since this research does not aim to provide the actual usability of such e-commerce website in the production environment, performance metrics, like scalability and throughput, are not our concerns.

3.2.2 Recommendation System

This component performs the two primary tasks of a typical recommendation system: retrieval and ranking. Retrieval is the process of preselecting a relatively small number
of items from a large pool based on a model or matrix constructed by product data, users' review data, or other types of information (Hron et al., 2020). Ranking is the rearrangement of the nominated items based on certain criteria, for instance, the highest estimated rating computed by the prediction algorithms (Hron et al., 2020). In this project, serving as a general recommender, the component is able to generate a list of recommended products for any existing user based on the reviews he gave in the past. That suggests, in this project, we do not seek to tackle the cold start issue in the recommendation system where the system does not contain the prior rating history of a certain user (Pandey & Rajpoot, 2016).

### 3.2.3 Chatbot

This component works similarly to the MongoDB chatbot mentioned in the related work section. Both chatbots are deliberately embedded in the associated websites. Whereas the MongoDB chatbot is designed for giving technical guidance to users looking for help, our chatbot Sophia focuses on providing e-commerce users with control over their own recommendation configuration and strategy. The following user control functionalities have been chosen for purpose. The first functionality is to allow users to be able to switch the recommendation algorithms, which is a representative user control approach that has been discussed and explored in several related works (Ekstrand et al., 2015; Harambam et al., 2019). We would like to see how it can be implemented in our infrastructure and how it fits the e-commerce setting. The second one is to let the number of recommended products be easily controlled by the user dynamically. This functionality may seem relatively straightforward but help to demonstrate how conversational systems
can influence the way that a user interface being designed by taking users’ opinions into account.

4 Methodology

In this section, we introduce how we implemented our infrastructure. Specifically, we discuss what the dataset, open-source projects, libraries, and frameworks were chosen, how we used them respectively, and how we integrated different components into a whole project. Moreover, we share the major problems we have encountered. The basic workflow and infrastructure diagram are shown in Figure 9.

4.1 Dataset

To build the proposed components, we adapted the Amazon product review data and product metadata of Amazon Review Data (2018) to create the data needed for our study. Amazon Review Data (2018) is the updated version of the Amazon Review Dataset released in 2014, containing product reviews and metadata from Amazon, including 233.1 million reviews spanning May 1996 - Oct 2018 (Ni et al., 2019). The whole dataset has 29 different categories of products in total.

In order to reduce excessive memory usage, we only picked the Clothing, Shoes and Jewelry sub-dataset with 32.2 million reviews and 2.6 million products as most of the e-commerce websites are related to fashion and clothing. In consideration of the data sparsity problem, we used the 5-core review data, which is a dense subset extracted from
the original product review data where each of the users and products has at least 5 reviews each.

Fig. 9. EWS’s infrastructure.
4.1.1 Data preprocessing

Nevertheless, even though the number of reviews in the 5-core data (11 million) is almost three times smaller than that of the total review data (32 million), the size of the 5-core data (5 GB) accompanying lots of string fields still make it impossible to load it altogether into the memory without using a cluster. Consequently, in order to handle the memory problem and obtain the data we need, we designed and implemented three different data parsers with respective purpose mainly using Python Pandas library on Google Colab, a free online cloud-based Jupyter notebook environment.

The first data parser reads both the 5-core review data file and the product metadata file by chunks and only keeps the necessary fields, i.e., columns, from each data file for saving memory. The chosen fields for the 5-core review data file are reviewer's ID, product's ID, rating, and Unix timestamp of that given review. The Unix timestamp is used by another parser for removing the duplicate reviews. The chosen fields for the product metadata file are the product's ID, brand name, title, and price. The resulting data files are significantly smaller with the memory usage of 2.3 GB in total.

The second parser is used mostly for data cleansing. It first read the two data files generated by the first parser and merged them together based on the product’s ID. Then, it dropped the reviews that contain null values in columns, like price and product title, and removed the duplicate reviews according to the product’s ID, reviewer’s ID, and review’s Unix timestamp. Besides, the parser removed several top reviewed brands that are either are not related to clothing or outliers having way more reviews than the rest of
brands in the dataset, for example, Converse. Lastly, as the whole merged dataset is significantly large, which may pose tremendous difficulty in training recommendation systems, we extracted the products of the top k most reviewed brands to construct several sub-datasets with different k values. This fits the overall design of our infrastructure as it is normal for the e-commerce website to group products by brands and only display certain brand’s products. This also makes our resulting dataset denser, given the fact that most of the reviews in the original dataset come from only a few popular brands. The results of each resulting dataset are shown in Table 1. The dataset with k = 20 is used for the rest of the research as its memory usage is moderate and containing enough amount of reviews and products.

Table 1. Resulting datasets with different k values.

<table>
<thead>
<tr>
<th>k</th>
<th>Number of reviews</th>
<th>Memory usage</th>
<th>Number of unique reviewers</th>
<th>Number of products</th>
<th>Average rating</th>
<th>File size</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>648353</td>
<td>262.2 MB</td>
<td>299473</td>
<td>4677</td>
<td>4.305</td>
<td>70 MB</td>
</tr>
<tr>
<td>10</td>
<td>1002432</td>
<td>400.7 MB</td>
<td>409257</td>
<td>7616</td>
<td>4.305</td>
<td>106 MB</td>
</tr>
<tr>
<td>20</td>
<td>1468379</td>
<td>621.2 MB</td>
<td>565228</td>
<td>13973</td>
<td>4.326</td>
<td>196 MB</td>
</tr>
<tr>
<td>30</td>
<td>1762809</td>
<td>820.8 MB</td>
<td>642848</td>
<td>18991</td>
<td>4.337</td>
<td>312 MB</td>
</tr>
<tr>
<td>40</td>
<td>1975163</td>
<td>906.2 MB</td>
<td>708067</td>
<td>22067</td>
<td>4.336</td>
<td>338 MB</td>
</tr>
<tr>
<td>50</td>
<td>2146013</td>
<td>1.1 GB</td>
<td>757756</td>
<td>26218</td>
<td>4.331</td>
<td>496 MB</td>
</tr>
</tbody>
</table>

Based on the dataset (k = 20), the third data parser then generated a product metadata dataset, which was loaded into the database of the e-commerce website later. Considering e-commerce websites normally present customers with the number of reviews and average rating along with other features of the given product, this parser computed and included these two values as fields for every product.
4.1.2 Statistics

The following figures show the distribution of ratings and the top 50 most reviewed brands in the whole merged dataset.

Fig. 10. Distribution of ratings.
Fig. 11. Top 50 most reviewed brands.

4.2 E-Commerce website

After exploring several open-source e-commerce projects with different tech stacks, we choose to build EWS based on an open-source website called Amazona (Jafarzadeh, 2020).

Compared to other alternative solutions, Amazona is optimal for our project in many aspects. To begin with, Amazona is the e-commerce demo created by its author, Basir Jafarzadeh, for one of his online web development courses. In the video tutorials, he basically explains every important detail and concept regarding to the development of a
basic e-commerce website (Jafarzadeh, 2020). Most of the videos are available on his YouTube channel for free and the whole demo is open-source. Secondly, one of the main purposes for that web development course is to teach viewers how to use the MERN stack, which is a web development framework consisting of MongoDB, ExpressJS, ReactJS, and NodeJS as its working components. Compared to other technology stacks for web development, such as the Spring Boot stack, because of its design and difficulty, MERN is very friendly to those who are unfamiliar with web development and extremely efficient for teams that need to build the prototype in a short period of time. Thirdly, the styling of Amazona’s web pages only relies on pure HTML and CSS. That means anyone knows the basic of front-end development can modify the looks of the webpages without the need to learn the other UI libraries, and thus reducing the study burden. We found that to be helpful as some of the user control options require altering the web pages in different degrees and using pure HTML and CSS with ReactJS makes that relatively convenient. Therefore, the excellent accessibility and design of Amazona make our project, EWS, easier to develop, more focused on user control in RS, and more comprehensible to other researchers and developers.

![E-Commerce with Sophia](image)

Fig.12. Home page of EWS.
For this project, the major changes that we made to Amazona are the following. (1) We changed Amazona from category-based to brand-based since our product data only have one category of products: clothing; (2) we imported the product data from the dataset generated by the data parsers into the MongoDB database and display them on the web pages successfully; (3) we created two different sections on the home page to display the top reviewed (i.e., popular) products and random products, which are implements by most of the e-commerce websites nowadays; (4) on the home page, we added the product recommendation functionality to the website by showing a list of recommended products, generated from the recommender, for a given user after logging in. The home page of EWS is shown in Figure 12.

4.3 Recommendation system

Without reinventing the wheel, we created our recommendation engine using Surprise, which is an easy-to-use open-source Python library famous for building and testing recommendation systems that deal with explicit rating data (Hug, 2020). The library implements 11 prediction algorithms for performing recommendation in total. In the context of RS, most of those algorithms are considered as a collaborative filtering technique, which is one of the most popular approaches to build a recommender that mainly based on historical data (Luo, 2019).

We evaluated most of the built-in algorithms using our rating dataset (k=20), while the k-Nearest Neighbors (KNN) algorithms were excluded due to their overwhelming memory consumption. First of all, we ran a 5-fold cross-validation procedure for all the
prediction algorithms to evaluate the accuracy and performance of each on the rating data. In detail, Root Mean Squared Error (RMSE) was used as our accuracy metric, and, for the prediction algorithms that rely on baseline estimates, the biases were estimated by using Alternating Least Squares (ALS). The results are shown in Table 2. Then, we picked the top 3 algorithms with the lowest RMSE (i.e., the highest prediction accuracy), including SVDpp, SVD, and CoClustering, along with the algorithm that has the worst accuracy, NormalPrecitor, for serving as the recommendation algorithm options that users can choose from. Lastly, as offline model training, we fitted the selected algorithms with the whole dataset (k=20) to train the models that can be used to predict the estimated rating for any given pair of user and product, and saved the models to pkl files.

Table 2. Results of prediction algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>test_rmse</th>
<th>fit_time</th>
<th>test_time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVDpp</td>
<td>0.837918</td>
<td>228.387429</td>
<td>9.627618</td>
</tr>
<tr>
<td>SVD</td>
<td>0.845430</td>
<td>90.262633</td>
<td>4.480360</td>
</tr>
<tr>
<td>CoClustering</td>
<td>0.934499</td>
<td>119.764880</td>
<td>4.218326</td>
</tr>
<tr>
<td>SlopeOne</td>
<td>0.953623</td>
<td>18.454979</td>
<td>7.509280</td>
</tr>
<tr>
<td>BaselineOnly</td>
<td>0.979175</td>
<td>9.116413</td>
<td>2.647920</td>
</tr>
<tr>
<td>NMF</td>
<td>0.990701</td>
<td>165.260995</td>
<td>3.219591</td>
</tr>
<tr>
<td>NormalPredictor</td>
<td>1.387144</td>
<td>4.028578</td>
<td>5.847533</td>
</tr>
</tbody>
</table>

In order to allow the e-commerce website to use these prediction models, we built a recommender server using Flask, a lightweight Python web framework. Along with providing some secondary services, such as listing all the available recommendation
algorithms and their information, the server mainly offers the product recommendation service based on the user’s ID using RESTful APIs. For any given user, the server firstly computes the estimated ratings for all product candidates with the trained model, deserialized from the pkl file, of either the default algorithm or the algorithm selected by that user explicitly. Next, the product candidates are sorted by their estimated ratings. Then, to represent the recommended products, a list containing the IDs of the top n products with the highest estimated ratings is sent back to the e-commerce website in JSON format. Both the variable n, which is the number of products being recommended, and the recommendation algorithm identifier are stored as parameters in the request sent from the e-commerce website and controlled by either the user or default setting. The default value for n is 4 and the default recommendation algorithm is set to be the one that has the highest prediction accuracy, which is SVDpp in our case. We regard these variables as part of the users’ personalized recommendation configuration and strategy.

4.4 Chatbot

After comparing many chatbot frameworks implemented by ReactJS, we selected react-chatbot-kit for building our chatbot Sophia (Fredrik, 2020). Similar to Amazona, the author of react-chatbot-kit made a series of video tutorials teaching how developers can build their own chatbot through this package, which we found to be very instructive. Even though this new chatbot project is still in development, its existing features and design are sufficient enough to meet the need of our project.
Written in ReactJS with pure HTML and CSS, this package is a reusable chatbot component that basically allows the developer to implement his own message parsers and associated actions that will be performed when certain message content being caught during the conversation. Thereby, for building a chatbot that aims to handle user control request for RS, we first came up with several keywords that are related to our user control methods or user control in general, such as “option”, “help”, “method”, “limit”, etc. Notice that the keyword “limit” stands for the number of products being recommended, just as the variable n mentioned previously. Then, we designed and implemented several message parsers that can not only detect those keywords but also determine what kind of responses should be returned to the user. An example of a response can be a text message telling the user which recommendation algorithm is being used currently. Instead of plain text messages, another type of response is React component being rendered when certain actions get triggered. In our design, these components, called widgets in react-chatbot-kit, are the fully customized interfaces that the user can use to view or change their personalized recommendation configuration. Serving as examples, two kinds of user control for RS we implemented are shown in Figure 13 and Figure 14. Both of them come with their own component and configuration, so, in terms of implementation, the user control functionalities are modularized to a great extent in our project.
Fig. 13. Switching recommendation algorithms.

Fig. 14. Controlling the number of recommended products.
It is worth pointing out that, same as adding the product recommendation functionality, we embedded Sophia in the e-commerce website incrementally. That is to say that the conversational system is added to the whole infrastructure by portions so that its development does not influence the other components and their functionalities.

5 Results and discussion

In this section, we discuss the results after we implemented and integrated the major components to give the readers an understanding of what our infrastructure is capable of doing by far. The results and their respective discussion will be presented by the order of the development.

First of all, after loaded with the actual Amazon product data we prepared, Amazona has been transferred into a more practical e-commerce mockup website. Significantly, this new website not only preserves most of the original services that Amazona provides but also becomes a data visualization tool that can be used to display a large amount of real-life e-commerce data from review or product datasets, such as the one we used: Amazon Review Data. In this project, more than 13 thousand unique Amazon products contained in our dataset (k=20) are ready for users to explore. Along with fundamental features like price and title, several important statistical values of a given product, such as the average rating and number of reviews, are also being displayed.
Moreover, as the recommendation system being implemented and connected to the e-commerce through its API server, our e-commerce website is now capable of recommending Amazon products for every existing user in our dataset. This new functionality makes the original website further closer to an actual e-commerce platform. More interestingly, given our infrastructure, the recommendation system is completely replaceable. That means we can use any other RS implemented by recommender frameworks or libraries other than Surprise, such as TensorFlow Recommenders (TFRS), to easily replace the current RS or let them collaborate with each other.

Most importantly, our chatbot Sophia embedded in the website is now capable of assisting the e-commerce customers to view or customize their personalized recommendation configuration and strategy. Through simple pre-designed conversation, product recommendation systems, which often being regarded as a black box, are now accessible to the customers in a conversational and interactive way yet still fully under the control of the e-commerce platform. Furthermore, no matter which extent of controllability the platform seeks to offer to its users, the implementation of any desired user control functionality using this chatbot can be easy, novel, and even more elegant than the traditional approach. Taking Amazon’s user control strategy for its RS that we mentioned previously as an example, instead of showing a long list of items purchased by a given user, as shown in Figure 3, a more user-friendly approach maybe is to ask the user a couple of simple questions that quickly guide the user to point out the item that he does not want to be used for recommendation. By this conversational approach, it is even possible to handle complex user requests regarding to user control for RS, for example,
“do not use the products I purchased or viewed last week for recommendation”, “do not recommend me products that often purchased by teenagers”, “no more Nike shoes”, “show my interested categories”, or “stop using my browsing history for recommendation”.

Last but not least, because of its easily understandable infrastructure and high extensibility, we considered this project might possess the potential to be turned into a general framework that can be used by other researchers to implement, present, and compare different user control functionalities or designs for RS. This proposed framework or solution is significant especially in terms of two aspects. First, even merely considering the e-commerce setting, there are numerous possible user control options for RS but not even one platform that can be used by the public to do the comparisons. Second, this public framework might be able to make the research in user control for RS be no longer limited to the actual e-commerce platforms, which helps more useful user control functionalities to be discovered.

6 Limitations

There may be several possible limitations in this study that could be addressed in future research. In this section, some major limitations and their proposed solutions will be identified and discussed as follows.

First, as the study focused on the user control for recommendation systems, we did not thoroughly evaluate our own RS implemented by Surprise. In detail, we only used
RMSE to evaluate the accuracy of our RS. However, besides the accuracy, there are many other metrics that can be used to evaluate an RS in various aspects, such as the relevancy metrics like recall and precision (Belhekar, 2020).

Second, our RS cannot handle the cold start issue, which means our system is not able to recommend products to the user who does not have any review data stored in our dataset beforehand. We plan to solve this issue in future research by adding a content-based recommendation system that recommends products based on a pre-computed matrix of product similarities and the browsing history of that given user.

Third, mainly due to time constraints, we could not conduct an online experiment to invite actual users for investigating whether performing user control in RS for e-commerce through the chatbot is more preferred than the traditional approach. Also, we did not explore which user control is valued by the users the most. These two tasks propose an important direction for our future studies.

Fourth, the conversation system is still underdeveloped. By only using pattern matching, we have not yet fully utilized the power of the message parser in our chatbot. In future studies, we will seek to build more intelligent parsers with Natural Language Processing techniques to provide Sophia a better understanding of users’ input and give users a more pleasant, convenient, and user-friendly experience.
Last, there are many details that can be improved regarding to our user interface. For example, we can find a way to show the alternative names of those recommendation algorithms that can be more easily understood by normal users. We can also display different kinds or versions of user control content according to the user’s identity or features.

7 Conclusion and future work

We propose an infrastructure to integrate an e-commerce website, a recommendation system, and a chatbot that aims to allow e-commerce customers to control their personalized recommendation configuration and strategy via simple conversations with the chatbot. Beside the future works mentioned in the previous section, some other works we plan to explore in the future are the following: (1) integrating the three different data parsers into a single parser for better usability, (2) adding equivalent user control functionalities by the traditional approach to the website for better comparisons, (3) improving several user interfaces on the e-commerce website, (4) increasing the size of our dataset, and (5) turning this project into a general framework that can be helpful and useful for other researchers or developers. We sincerely hope that, with its unique mission, one day our chatbot Sophia could help virtual personal assistants to be more accepted by the e-commerce users in order to take everyone's online shopping experience to the next level.
References


