San Jose State University

SJSU ScholarWorks

Master's Theses Master's Theses and Graduate Research

Spring 2022

SMARTREC - A Smart Conversational Recommendation System Using Semantic Knowledge Graphs

Sudha Vijayakumar
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation
Vijayakumar, Sudha, "SMARTREC - A Smart Conversational Recommendation System Using Semantic Knowledge Graphs" (2022). *Master's Theses*. 5281.
DOI: https://doi.org/10.31979/etd.r5ja-qnf2
https://scholarworks.sjsu.edu/etd_theses/5281

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

SMARTREC - A SMART CONVERSATIONAL RECOMMENDTION SYSTEM USING SEMANTIC KNOWLEDGE GRAPHS

A Thesis

Presented to

The Faculty of the Department of Computer Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Sudha Vijayakumar

May 2022

© 2022

Sudha Vijayakumar

ALL RIGHTS RESERVED

The Designated Thesis Committee Approves the Thesis Titled

SMARTREC - A SMART CONVERSATIONAL RECOMMENDATION SYSTEM USING SEMANTIC KNOWLEDGE GRAPHS

by

Sudha Vijayakumar

APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING

SAN JOSÉ STATE UNIVERSITY

May 2022

Jorieta G. Jetcheva, Ph.D.	Department of Computer Engineering
Mahima Agumbe Suresh, Ph.D.	Department of Computer Engineering
Magdalini Eirinaki, Ph.D.	Department of Computer Engineering

ABSTRACT

SMARTREC - A SMART CONVERSATIONAL RECOMMENDATION SYSTEM USING SEMANTIC KNOWLEDGE GRAPHS

by Sudha Vijayakumar

CRS strives to return the most relevant recommendations to users through a multi-turn interactive conversation. The Covid-19 pandemic has undoubtedly accelerated the pace of adoption of conversational AI systems by many online platforms needing to provide highly available customer support. This ongoing demand calls for the need to implement more generic and efficient conversational agents and recommendation engines that can provide customers with the required information at every stage of the user's interaction with the e-commerce platform. This study presents a Smart Conversational AI-based recommendation system - SMARTREC, which enables a multi-turn conversation with the user to understand the context and semantics behind their product requirements and generate appropriate recommendations in real-time. Several ongoing studies investigate how to evolve CRS, there are many open research problems in this space. Current CRS suffers from four major issues. First, a lack of proper contextual understanding of the user intention; second, inaccurate semantic mapping of user preferences in natural language to the interested item attributes; third, rely only on current conversation and suffer from data sparsity; fourth, trained on open-domain crowd-sourced conversational data preventing the system from learning the user intentions accurately. This study implements a novel real-time CRS by curating large-scale domain data, further combining them with a common-sense semantic network to build an intelligent domain knowledge graph. Finally, this study conducted extensive experiments to demonstrate the efficiency of SMARTREC in yielding better performance as a CRS.

ACKNOWLEDGEMENTS

I want to express my gratitude to my advisor, Dr. Jorjeta Jetcheva of the SJSU Department of Computer Engineering, for providing me with excellent guidance and themotivation to finish this study. Her unwavering support, patience, and advice inspired meto perform a thorough investigation throughout the study, experimenting, and writing stages of this work.

I want to thank my committee members, Dr. Magdalini Erinaki and Dr. Mahima Agumbe Suresh, for the knowledge and ideas I acquired through the coursework, which issignificant in completing the thesis.

My sincere reference to Professor. Marie highby and Cheryl R. Cowan, SJSU Graduate Studies Associate, were tremendously helpful in guiding me with the writingrequirements and submissions.

Finally, I would like to acknowledge my family for their continued support andencouragement to continue my aspirations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI - Artificial Intelligence
ALS - Alternating Least Square
CBF   Content-based filtering
CF - Collaborative filtering
CMPE - Department of Computer Engineering
CONF - Conceptual filtering
CRS - Conversational Recommendation System
CS-DKG - Common-Sense Domain Knowledge Graph
CTR - Click Through Rates
CVR - Conversion Rate
DKG - Domain Knowledge Graph
ESCIN - Exact, Substitute, Compliment, Irrelevant, No Results
FAQ - Frequently Asked Question
LaTeX - Lamport TeX
MAR@K - Mean Average Recall for top-K recommendations
NER - Named Entity Recognition
NLG - Natural Language Generation
NLP - Natural Language Processing
NLU - Natural Language Understanding
SJSU - San Jose´ State University

1 INTRODUCTION

According to recent data, 72 percent of e-commerce users seek answers online [1]. By 2022, according to Gartner, 70 percent of middle-class workers will utilize conversational agents each day [2]. 58 percent of consumers care even more about the customer experience in the post COVID world. Therefore, Conversational Recommendation Systems (CRS) are gaining popularity as a next-generation enabler of e-commerce systems. A CRS is a natural medium for users to interact with e-commerce platforms. It mimics the way users interact with customer support personnel or a knowledgeable friend in everyday life. The need for CRS became even more pronouncedduring the unprecedented COVID-19 pandemic when customer support systems becameunavailable and left customers unable to ask questions or resolve issues leading to widespread deployment of basic chatbots [3]. The Covid-19 pandemic has undoubtedly accelerated how e-commerce platforms adopt conversational agents for highly available customer support, and Gartner forecasts support that hypothesis. This projected demand calls for the need to implement generic and efficient conversational agents and recommendation engines to provide customers with the required information at every e-commerce interaction, from browsing to purchasing to the issue-resolution cycle. The need for a CRS [4] has triggered the emergence of research efforts around capabilities to provide high-quality recommendations through interactive discussions with users. The ability of the conversational recommendation system to allow users to express interests in natural language can dramatically improve e-commerce operations.

This work implements a CRS using publicly available large-scale Airbnb data [5]. The large-scale Airbnb data presents a substantial modeling challenge for our proposed study and represents a broad range of use cases involving consumer choices and decisionsupport. Furthermore, using the large-scale Airbnb data to train the system allows us to evaluate how well models can perform in real-life customer support applications, especially in similar e-commerce or vacation planning applications. A good product recommendation or customer support system relies on an accurate interpretation of the user preferences. Therefore, we propose to leverage a semantic merging approach to contextually map the user preferences to the available products or suggestions while remembering the user preferences during the conversation. A word-oriented knowledge graph (i.e., ConceptNet5 [6]) merged with Airbnb's social conversational data and itemattributes. An item-oriented knowledge graph adds data intelligence to the system using Airbnb's item listings, historical user interactions, and user reviews. This study will implement an integrated recommendation engine using a dialog component [7] that willact as retrieval and generative response model to list the relevant recommendations and response text. Our generic solution architecture for AI-enhanced product recommendationand customer support systems in e-commerce can be easily applied to other datasets and domains.

One of the technical challenges behind CRS is that it requires a coupling between the recommender and dialog parts. The dialog explains user intentions and answers to past expressions with reasonable responses. Then the recommender part learns user inclinationand suggests closely matched items dependent on context-oriented natural language expressions. Traditional conversational recommendation systems [8], [9] incorporatestwo core components: 1) a language model to validate partially formatted user requirements and 2) a switching controller to choose between the dialog component andthe recommender component. Despite some advances in this area, two significant issuesstill need to be addressed. First, a discussion comprises a couple of sentences and thus lacks adequate, relevant information for precisely understanding user inclination.

Capturing user interests through interactive conversation in natural language is a tedious task; second, real user intents mainly depend upon the facilities they sought or experiences/emotions driven by ordinary expressions during the conversation [10]. So, it is natural that a semantic gap exists between the user requirements required to provide recommendations and the way items are defined [11]. For example, consider a case when the user searches for a place with amenities to keep the food unspoiled, a peaceful locality with a hill view where the item attributes are defined differently with keywords like refrigerator, mountain house with quiet ambiance. Therefore, a semantic mapping of the user's natural language expressions to the product definitions is crucial for generating good recommendations. Say a user is searching for peaceful and quiet Airbnb places like "Serene ocean side house," where fewer sentences portray the user's intention. The context of the conversation has to be understood to capture the user's interests accurately.This example is essential to capture the hidden desire for a "safe" lodging and recommend the Airbnb posting "Serene ocean-side house." As illustrated in Fig. 1, our framework shows the user both the suggested item listing(s) and the justification/explanation behind its recommendation(s). Without bridging the semantic gap, it is impossible to determine the user intent and arrive at relevant suggestions. For example, consider a case where an item listing with the description "amenities includes refrigerator" cannot be mapped to the user

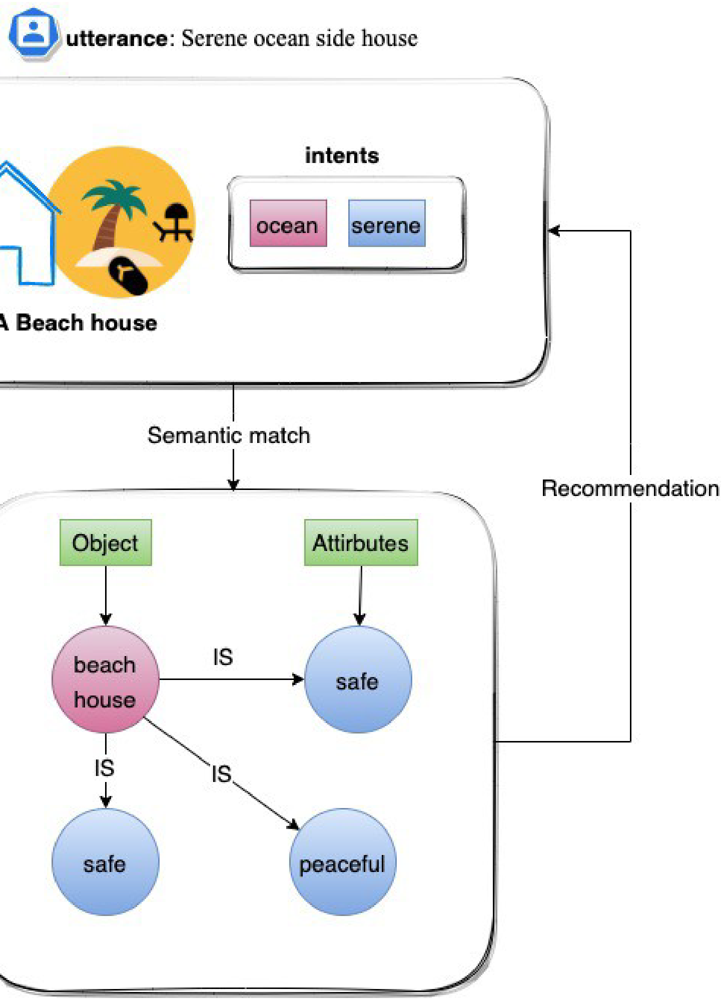


Fig. 1. How information is mapped? - Human vs AI.

intent in natural language expression such as ”I am looking for amenities to keep food cold” by default.

Current CRS [12]-[16] focus mainly on gathering all the item information to extract a particular item from the item space, disregarding the word-levelvariations (e.g., semantic connection between the texts, safe and serene). Besides, existingworks have not considered the semantic gap between user language and item-level information. Accordingly, they do not fully utilize the available information in a Knowledge Graph. The issue starts from the point when dialog and recommender parts cannot coordinate appropriately due to a lack of understanding of how the user describesan item and how item attributes are defined. To address this shortcoming, we merge the item attributes in a semantic item-based knowledge graph with ConceptNet5 [6] to create a common-sense semantic knowledge graph that enables us to recognize these mappings. Furthermore, we provide an interface with Conversational AI-based languagemodels for mapping user requirements to the available items, thereby coming up with semantically and conceptually meaningful item recommendations.

This study presents SMARTREC, a Smart Conversational AI-based recommendation system. SMARTREC can have a multi-turn conversation with the user to understand the context and semantics behind their product requirements and generate appropriate recommendations using natural language responses. Unfortunately, most of the state-of-the-art CRS are: (1) trained on crowd-sourced conversational data and lack variety, (2) lack meaningful multi-turn conversation, (3) lack contextual understanding ofthe user preferences, and (4) possess a semantic gap between the way the user search fora particular product and the way product attributes are stored by the e-commerce site.

This study proposes a generic and re-usable solution architecture that has the following core components: (1) language models that capture the context from the currentconversation and query the knowledge graphs for relevant products, and (2) an integratedrecommendation and dialog engine to generate product and solution recommendations using the current user interactions. The goal is to merge current data from the user conversation and historical user-item interactions of a particular domain with ConceptNet5 [17] to understand the user requirements, match them better, and return theproducts/recommendations that interest them.

This study evaluates the proposed approachin the context of an open-source Airbnb dataset, combined with an auto-curated conversational dataset. We have shown that it performs well in both recommendation andconversational tasks in the e-commerce domain.

More specifically, SMARTREC implements: 1) an NLU pipeline to extract user intents from the current conversation, 2) a common-sense knowledge graph (ConceptNet5) integrated into the domain knowledge graph (Airbnb item listings, reviews), enabling an extended search of related concepts against the product attributes, 3)a real-time recommendation engine built using semantic knowledge graph with an explainable AI(where recommendations are tagged with related concepts explicitly to carry out the extended search) 4) auto-generated user intents from actual user reviews.

The key contributions of this research are:

A fully automated data gathering and transformation pipeline for curatedconversational data.

A semantically trained machine learning model captures the context of the userintent in natural language.

A real-time recommendation framework using common-sense semantic knowledge graphs to fetch Airbnb recommendations.

An comparative evaluation of our proposed system relative to existing state-of-the-art systems, based on time-to-train the machine learning models, dataloading times, and query response times.

The organizations of the remainder of the thesis chapters are as follows. Chapter 2, will discuss in detail all the related works. Chapter 3 will discuss the high-level functionalities and expected output from the system. Chapter 4 will discuss the solutionarchitecture and the proposed approach in detail. Chapter 5 will briefly discuss the experiments conducted and results obtained, Chapter 6 will highlight the future work plans for this thesis, and Chapter 7 will discuss the conclusions drawn.

2 RELATED WORKS

A typical architecture of a conversational AI-based recommendation system (CRS) shown in Fig. 2 consists of two core components, namely the recommendation engine for generating appropriate recommendations and a language model for understanding user queries, matching with appropriate action endpoints, and developing responses.

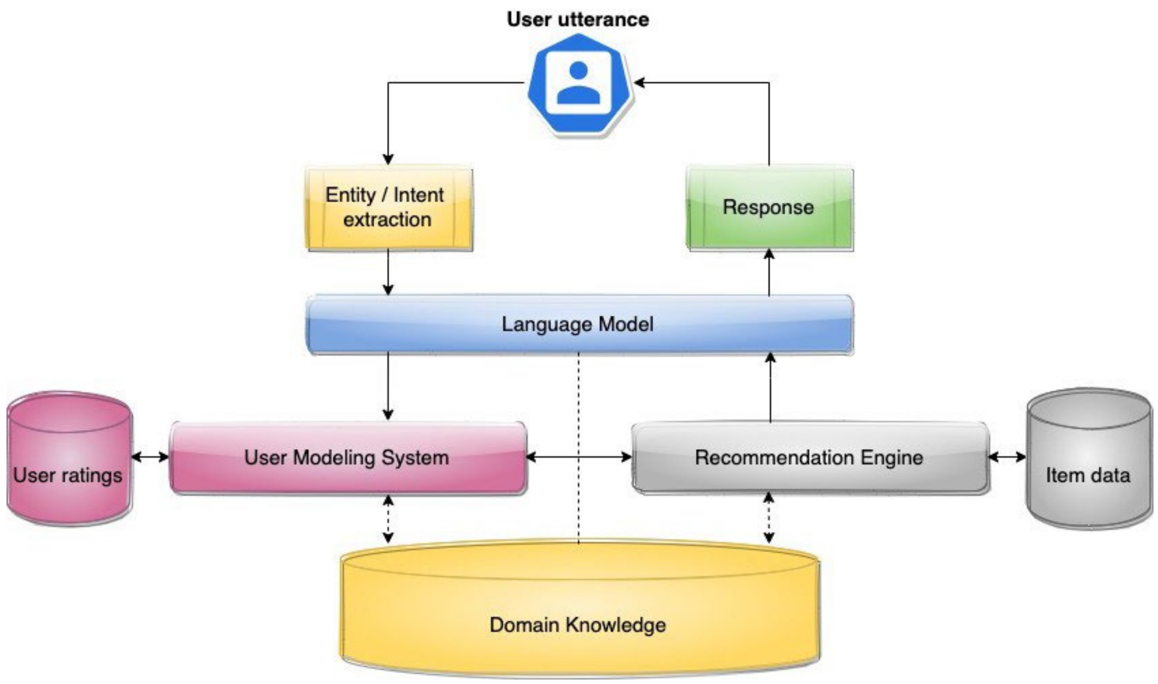


Fig. 2. Typical architecture of a conversational recommendation system.

Recommendation frameworks extract a part of the itemset from the dense item space defined by item attributes that satisfy the user requirements. Traditional recommendation engines consider the past user ratings (e.g., rate, click, number of views, buy) for the itemsets to capture the user interactions in the item space; this, in turn, is considered to estimate the user alignments with a different set of items. Be that as it may, user-item collaboration information usually is meager. Therefore, recommendation engines incorporate numerous methods to tackle information sparsity issues, such as using side-information such as user surveys and scientific categorization information to compute user requirements implicitly. CRS predominantly centers around the recommendation setting through current discussions rather than authentic collaboration information as a comparative approach. Remarkably, many novel recommendation solutions are beginningto embrace knowledge graphs to enhance the performance of the recommendation task bygetting explicit user requirements and applying semantic logic of the explainability to retrieve exciting items. Building semantically meaningful knowledge graphs is a challenging task. It involves mapping the user utterances conceptually equivalent to the interested product attributes defined by the rigid schema. The below section will discusssome interesting related works in recommendation frameworks and language modeling for capturing the context behind user utterances and highlight the pros and cons of each method.

2.1 Generative and Retrieval-Based Models

Conversational AI frameworks aim to generate appropriate responses for the given multi-turn context-oriented expressions. Existing works can be classified either into generative or retrieval-based methods [12] where the generative-based approaches use learnable models to create the response text. In contrast, the retrieval-based methods attempt to develop the semantically meaningful response from a vast store of historical user discussions. With the improvement of sequence-sequence language processing transformer models, various architectural extensions aid in understanding language utterances and generating practical responses. However, training on language models andbuilding the offline embeddings as part of the training process is a computationally intensive, time-consuming, and routine task to be on par with new training data. Again, whether generative or retrieval-based methods, both interface with the recommendation frameworks in collaborative or content-based approaches where the data sparsity issue still holds to generate appropriate recommendations. Moreover, both the methods rely upon the historical user and item interactions without a flexible contextual match with the current conversation. So, there is a logical gap in how such frameworks can map userrequirements to the right items.

2.2 Neural Collaborative Filtering Models

Traditional conversational recommendation engines essentially used predefined functions to communicate with users. However, as of late, a few investigations have begunto coordinate the two parts for decoding user intentions and suggest the appropriate itemsthrough regular natural language articulation. Generally speaking, these techniques underscore the detailed proposal, while straightforward or heuristic arrangements carry out the conversation. In most cases, a crowd-sourced conversational dataset trained on deep neural network models generates appropriate responses.

Neural collaborative filtering models [14], [15] introduce a neural architecture replacing the inner product on the latent user and item features in the traditional collaborative approaches. It further implements a multi-layer network to learn the user-item interactions while generalizing on the matrix factorization and proves that adding more profound layers of neural networks offers better performance on the recommendations. However, deeper neural models are very complex to configure, computationally intensive, and present substantial challenges to offline training using GPUs. Also, neural networks are awell-known black-box training methodology, and explainable facts cannot support the recommendations. Neural models trained on much general crowd-sourced conversationaldata could deviate from the domain's original conversation. Moreover, further examinations of such methods joined diverse Knowledge Graphs to work on the CRS. The main focus was to improve the identification of items. Semantic Knowledge Graphsare used to stitch together corresponding data from different sources in large-scale distributed systems.

2.3 Knowledge Graph-based Semantic Fusion

KGSF [11] proposes a unique conversational recommendation system by incorporating and combining a common-sense ConceptNet knowledge graph with an available domain item-level DBPedia knowledge graph [18]. KGSF also relies upon the deeper neural network models, excludes the historical user-item interactions to train the recommendationand language models, and represents the knowledge graphs in the simple file system. Bymerging knowledge graphs, the KGSF models can cohesively learn data representations by maximizing the mutual information for better recommendation and response generation. However, generating and combining million-scale semantic embeddings of domain knowledge graphs with a file-based representation is not scalable. It requires hours of data preprocessing and days of intensive training to train the models fully.

Furthermore, it could take even more days to train the model when the training includes the historical user-item interactions presenting the need for a high-power computing system like GPU to carry out machine learning experiments. Also, usingavailable domain data can add too much variance to the resulting model, and crowd-sourced conversational data could bias the overall result.

2.4 Towards Conversational Search and Recommendation: System Ask, UserRespond

This paper [2] presents a MultiMemory Network (MMN) architecture trained on large-scale multi-domain datasets for e-commerce. The core of any conversational recommendation system is to allow users to have a free-style conversation more naturallyand return recommendations that at least match conceptually close to the requirement.

However, this system asks intent-based queries in a sequential order to develop proposals, thereby restricting a free-flow interactive conversation. This system also carries out a personalized search, and the results returned are based on threshold scoring criteria. Likeany other approaches presented above, this work also relies upon offline training of neural networks. Therefore, it could suffer from similar challenges like more time to train the models, the requirement of high power computing, and explainable facts that cannot support results.

2.5 Billion-Scale Commodity Embedding for E-commerce

This paper [13] addresses three significant issues with the recommendation task: scalability, data sparsity, and cold start issues. First, this method implements a graph embedding framework. Graphs constructed from the user-item historical interactions anditem set, alongside graph embeddings, are generated to compute the pairwise similarity between the item group. This approach addresses cold start problems and sparsity issuesby incorporating side information and user-item interactions. Offline experiments and A/Btesting through Click-Through-Rates(CTSs) demonstrate better performance over collaborative approaches. This method still needs to train on a billion-scale data offline, presenting the need for hour-long intensive training using high computing power. The choice of side information included improving cold start and sparsity could impact the way recommendations are generated and not explainable. Finally, both offline and real-time experiments are required to conclude the results achieved, which is not a flexiblearchitecture to adapt.

Based on this past research, we propose designing and implementing a novel conversational recommendation system that uses item-based semantic knowledge graphs merged with common-sense knowledge graph embeddings in real-time. The proposed work will automatically auto-curate the conversational data using the available item entities in the user reviews retrieved from the predefined user profile. Furthermore, the study will implement an NLU and an NLG to extract entities, identify user intents from the user interaction, and generate appropriate responses. The recommendation part of theproposed solution will depend upon a [8] real-time collaborative, content-based and contextual (relevant search) filtering approaches [19]. Overall, we present a highly scalable semantic mapping of the item-set, conversational data with a common-sense knowledgegraph framework allowing for real-time in-DB NLP and explainable recommendations returning the response within seconds.

3 PRELIMINARIES

SMARTREC aims to recommend Airbnb item listings to the end-user by having an interactive multi-turn conversation:

RASA [20] decodes the user requirements in the natural language during the conversation to understand their preferences.

It conducts a common-sense induced semantic search in a pre-built knowledge graphto come up with real-time recommendations.

RASA [20] continues the conversation to get relevant recommendations or asks formore clarifications if there are no suitable matches.

SMARTREC implements two core components: the RASA conversational AI [20] interface as the front-end and a real-time recommendation framework using the Neo4j [21] graph database as the backend. The dialog component communicates with thereal-time recommendation engine APIs through the RASA action endpoints. A better understanding of the user expectations and appropriate recommendations for Airbnb itemlistings is the overall goal for this system.

Methodically, the Neo4j recommendation engine extracts a set of Airbnb item listings from the entire item listings using collaborative filtering, content-based filtering, or conceptual filtering (relevance search). At the same time, the RASA Conversational AI [20] solution generates natural language responses to return the recommendationsfetched to the conversing user.

4 SOLUTION ARCHITECTURE

The proposed solution shown in Fig. 3 will leverage the RASA conversational AI development platform as the front-end language model to handle the interactive conversation, natural language understanding, entity extraction, and generate natural language responses back to the user. The RASA solution consists of four core components as follows: 1) an NLU model trained using sample user intents to infer theuser intents from the interactive conversation, derive the context and entities from the natural language expressions, 2) a Dialog State Manager implemented through RASA stories will keep track of the conversation flow, and trigger corresponding action pointsfor every user intent specified and execute the chain of actions through custom actions and, 3) custom RASA actions to query the real-time semantic knowledge graphs API using the identified entities from the interactive conversation and generate natural language responses which is a set of Airbnb recommendations back to the user.

The backend recommendation system comprises four different components: 1) a continuous data pulling component, 2) a data preprocessing component, 3) a set of data generation components using pretrained machine learning models, 4) Neo4j cypher scriptsto load data and merge common-sense ConceptNet5 knowledge graph into the Airbnb domain knowledge graph and 5) a real-time recommendation engine API implemented onthe top of the semantically combined knowledge graphs.

The following sections will discuss the internal functionalities of the components and demonstrate the input and output from every component interface.

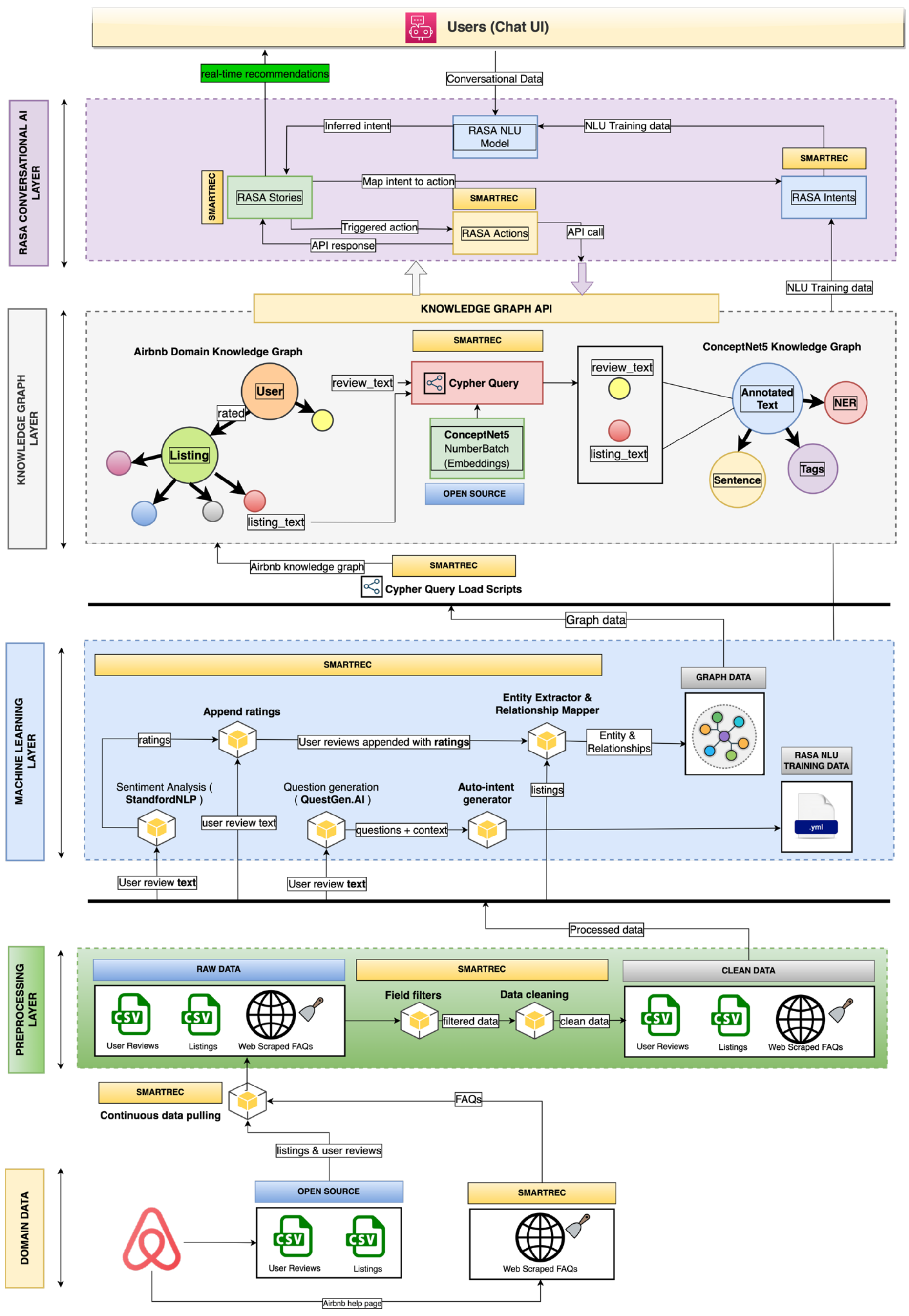


Fig. 3. SMARTREC Solution Architecture.

4.1 Domain Data

As shown in Fig. 4 below, this solution uses the two types of data related to theAirbnb domain as listed below,

Open-source Airbnb item listings and user reviews [5].

FAQ links from the Airbnb official help page [22].

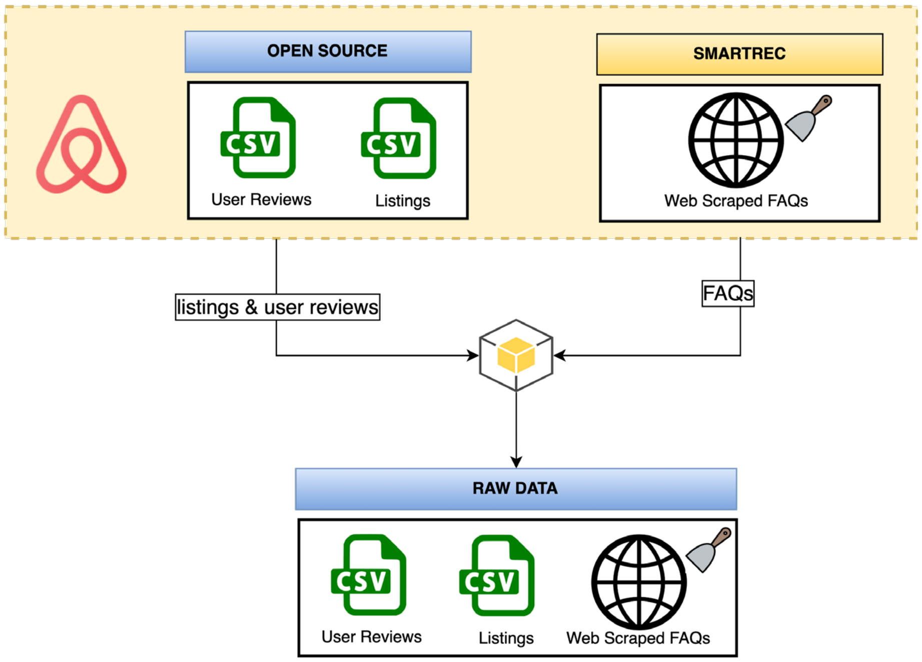


Fig. 4. Domain data - Automated data pulling.

SMARTREC implements a set of data pulling scripts to automatically download the Airbnb opensource data and web scrap the Airbnb help page for FAQ help links. There are 5402 item listings, approximately 310K user reviews downloaded for the ’Amsterdam’ location, and around 81 help links scrapped using the scripts. The scripts can be easily modified and extended to include multiple Airbnb locations and fully automated to implement a continuous data integration of global Airbnb locations. This solution will construct a semantic domain knowledge graph to evaluate the proposal using the downloaded item listings, user reviews, and FAQs.

4.2 Preprocessing layer

As shown in Fig. 5 below, SMARTREC preprocesses the raw data downloaded in the previous step, which includes the Airbnb item listings, user reviews, and FAQs as follows:

Feature selection for the item listings data using a dynamic configuration file.

Append a new feature called 'listing text' in the item listings data. Concatenating the attribute values of every record generates the item listings text field. The listing textwill be expanded into concept tags using the common-sense ConceptNet5 semantic graph.

The text is further processed to remove stopwords, punctuation marks, special characters, numbers, and finally converted to lowercase.

4.3 Machine learning layer

As shown in Fig. 6 , SMARTREC implements a machine learning pipeline toperform the below steps:

Generate User rating: The open-source Airbnb dataset contains only individual user reviews for every item listing and missing the individual user ratings. Therefore,user rating generation must perform collaborative-recommendation filtering to compute item-item similarities based on user ratings. The pre-trained StandfordNLP [23] machine learning model is used to carry out the Sentiment analysis on the user review text to generate user ratings on a scale of 1 to 5.

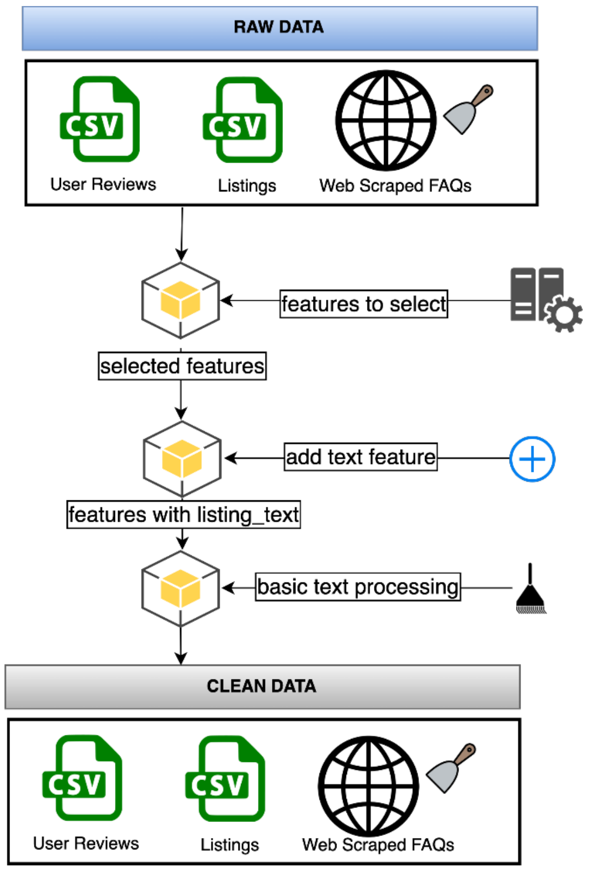


Fig. 5. Preprocessing layer.

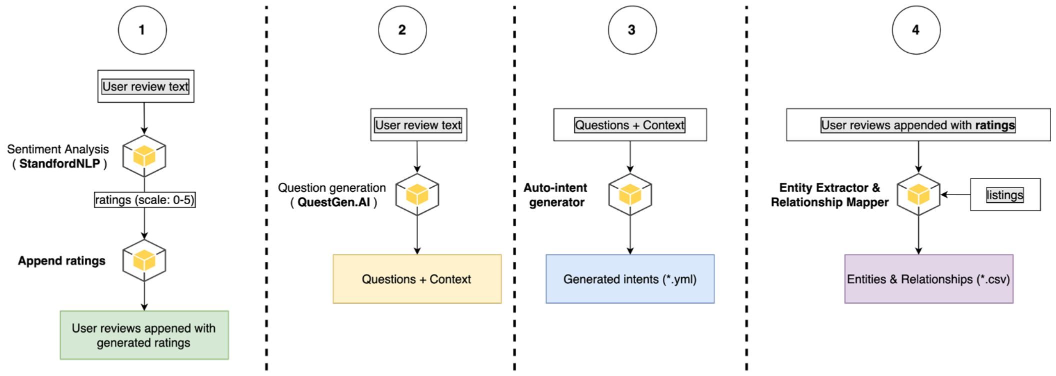

1
User review text
Sentiment Analysis (StandfordNLP)
ratings (scale: 0-5)
Append ratings
User reviews appened with generated ratings
2
User review text
Question generation (QuestGen.AI)
Questions + Context
3
Questions + Context
Auto-intent generator
Generated intents (*.yml)
4
User reviews appended with ratings
Entity Extractor & Relationship Mapper
listings
Entities & Relationships (*.csv)

Fig. 6. Machine learning layer.

Question generation: This step involves the generation of questions automatically from the cleaned user review text using the pre-trained QuestGen.AI [24] machine learning model. QuestGen.AI generates a triplet (question, answer, context) for everysentence within a review text paragraph.

RASA intent generation: This step involves the automatic generation of the NLU training data for the Rasa NLU model. The NLU model file nlu.yml is populated automatically with intents using the context and the intent examples using the questions from the triplet(question, answer, context) generated in the previous step.

Extract entities and relationship: This step involves the generation of unique entities from the clean FAQs, user reviews, and item listings data using automatedscripts. Extracted entities included users, FAQs, item listings, amenities, beds, property type, room type, listing text, review text, etc. In addition, this step also involves the extraction of relationships between different entities and the automaticgeneration of corresponding relationship mappings. Extracted relationships include'rated', 'has room type', 'has property type','has amenity','has beds','has listing text','has review text','has bedrooms' etc.

4.4 Knowledge Graph layer

As shown in Fig. 7, SMARTREC implements cypher queries to execute the following steps:

Construct semantic domain knowledge graph: Bulk-upload the processed Airbnb domain data into the Neo4j Graph Database. After loading, there are 247,376 nodes and 418,817 edges created in the Neo4j graph database.

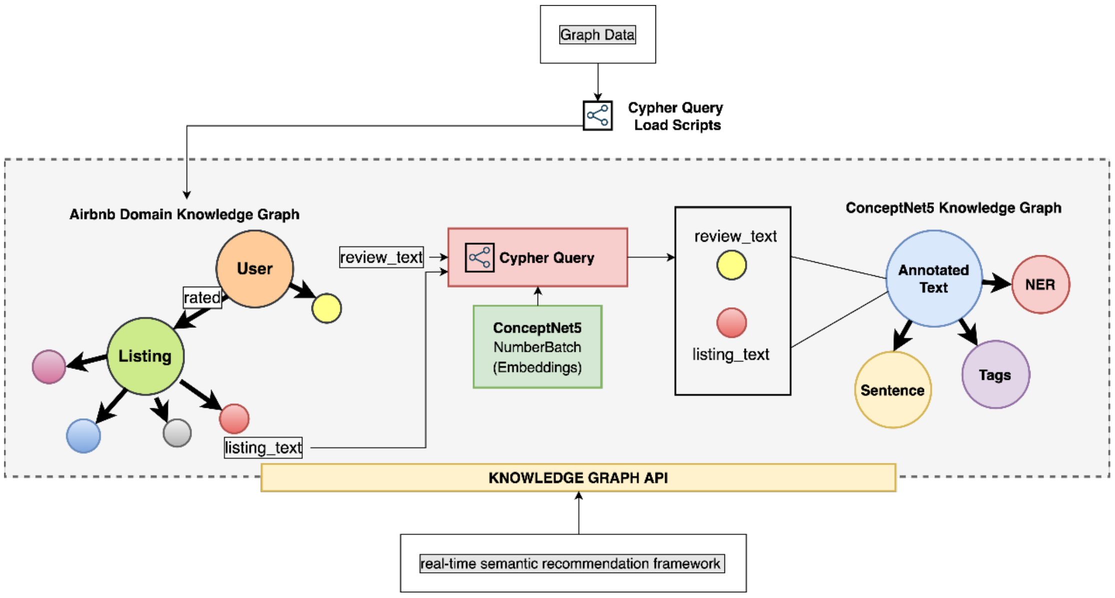

Graph Data
Cypher Query Load Scripts
Airbnb Domain Knowledge Graph
User
rated
Listing
listing_text
review_text
Cypher Query
ConceptNet5 NumberBatch (Embeddings)
review_text
listing_text
ConceptNet5 Knowledge Graph
Annotated Text
NER
Sentence
Tags
KNOWLEDGE GRAPH API
real-time semantic recommendation framework

Fig. 7. Knowledge graph layer.

Merge common-sense knowledge graph: Merge the word embeddings of the ConceptNet5(ConceptNet Numberbatch, language: en) [6] common-sense knowledge graph into the review text and the listing text entities. After merging, thetotal number of nodes increased to 12,443,574, and the number of edges increased to 36,193,223. The newly generated edges include expanded nodes such as concepttags, NER, etc., of the review text and the listing text entities.

4.4.1 Knowledge Graph API

After constructing the domain knowledge graph and merging the ConceptNet5 common-sense knowledge graph, Neo4j exposes a real-time semantic querying API [25],with which the RASA conversational AI platform interfaces. This solution presents a real-time in-DB querying recommendation framework to fetch recommendations using: 1)item-based collaborative filtering, 2) user-based collaborative filtering, 3) content-based collaborative filtering, and 4) conceptual filtering - relevant search using concept tags from the common-sense knowledge graph as shown in Fig. 8. Furthermore, this solutioneliminates offline training to build the item-item or user-user vector space to compute similarity and make recommendations.

4.5 RASA Conversational AI layer

This study as shown in Fig. 9 will use the opensource RASA Conversational AI platform to implement an interactive chatbot for the users to ask for Airbnb recommendations. The RASA framework [20], [26] provides many custom options to include goal-oriented datasets and allows flexibility to add multiple APIs. RASA handlesa conversation using two models as follows:

An NLU model [27] decodes a user utterance to identify the user intents and derive entities. RASA provides different choices of predefined RASA-provided or custom configurations to choose between featurizers, entity extractors, and intent classifiers.

A RASA Core model [28] to route calls to the custom actions for the classified intents generates appropriate responses for the current user utterance. The Core model keeps track of the current state of the conversation through stories defined anda set of machine-learning-based policies to choose appropriate responses from the domain configurations or the custom actions

.

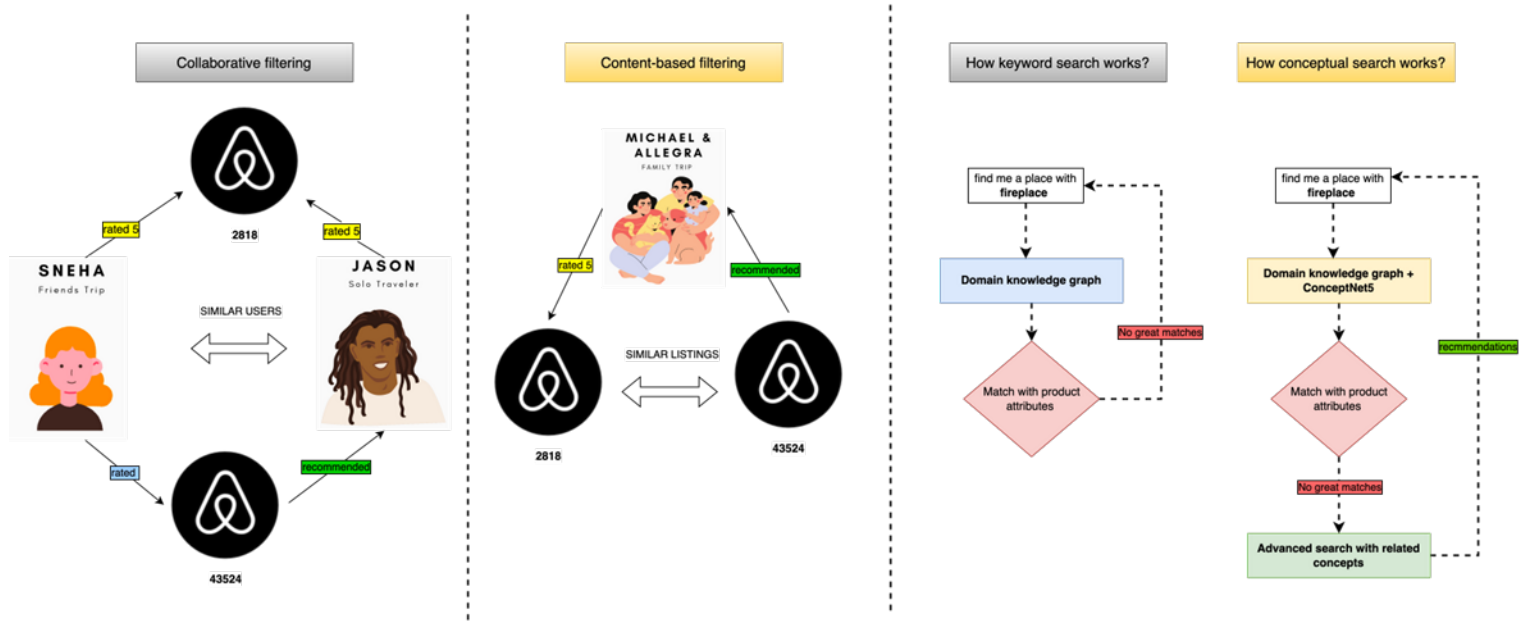


Fig. 8. Different recommendation approaches.

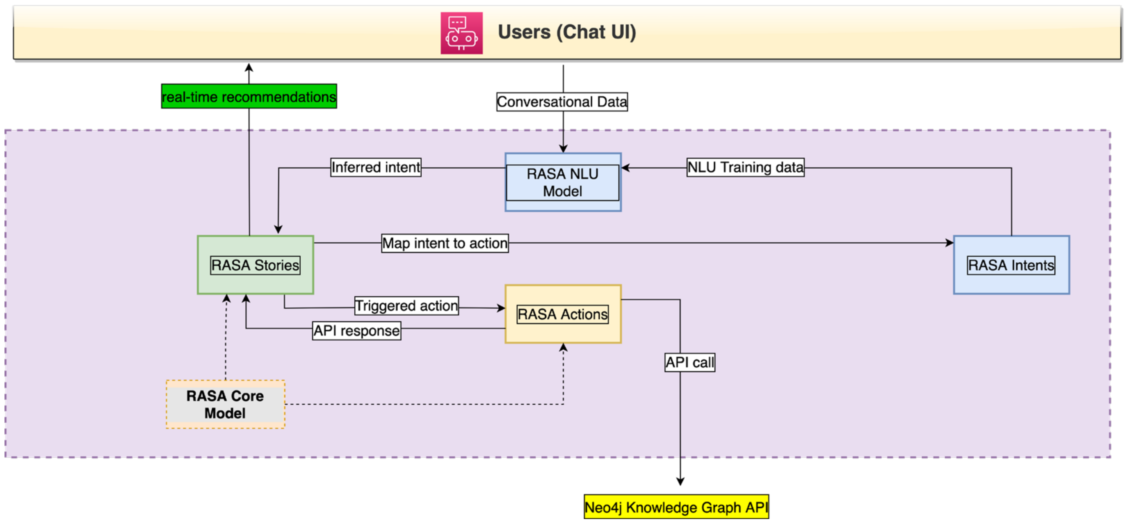


Fig. 9. RASA Conversational AI layer.

As show in Fig. 10, the RASA Conversational AI architecture [31] processes an interactive conversation as follows:

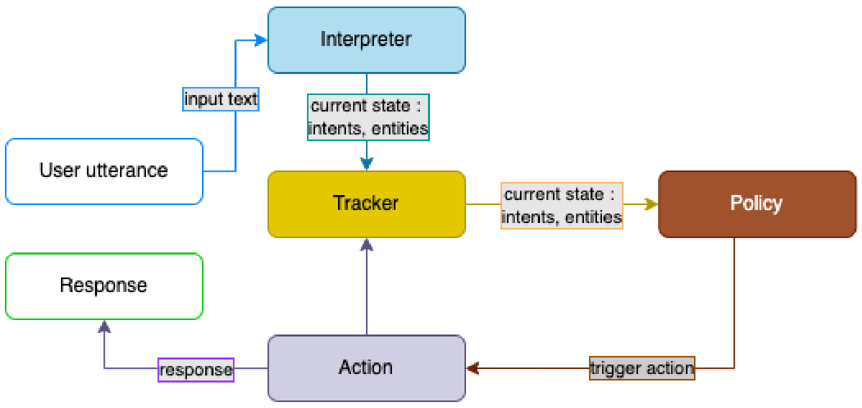


Fig. 10. RASA Conversational AI process flow.

An interpreter converts the input message into a key-value pair consisting of theraw text mapped to the corresponding intents ad entities detected.

A tracker keeps track of the current state of the conversation.

A policy learns the current state using the tracker object.

The policy decides the action to be triggered depending upon the learned state.

The tracker object updates the current state of the conversation by logging theaction triggered in the previous step.

RASA core generates a response to be returned to the current conversation.

This solution also avoids the cold start problem by conceptualizing the user requirements in the knowledge graph [29]-[32] even when fewer data are available to compute similar items or users. The merging of the commons-sense knowledge graph into the domain knowledge graph provides a comprehensive search framework where exact user requirements are matched to related concepts to fetch meaningful recommendations as per the user requirements. For example, the user utterance is ’find me a place close to tramlines.’ Regular keyword-based search will try tomatch the entity tramlines in the item listing attributes. If no exact matches are available,the system shall return empty results. In contrast, SMARTREC will conduct an extendedsearch for all the related concepts like trams, transport systems, tramcars, etc., increasingthe chances of closely matching item listing recommendations.

5 EXPERIMENTS AND RESULTS

This section will detail the experimental methodology for SMARTREC.

5.1 Experimental Setup

5.1.1 Dataset

The proposed real-time recommendation framework and the RASA conversational AI solution are evaluated using the processed Airbnb domain data [5], [22] summarized in Table 1. In addition, the auto-generated user intents are used as the conversational datato train the RASA NLU model. Finally, the processed Airbnb user reviews and item listings are translated into a domain knowledge graph, and ConceptNet5 Numberbatch [17] embeddings are merged into it to create a common-sense domainknowledge graph in a real-time recommendation framework setup.

Table 1
Data Summary

Type	**Size**
Raw Dataset	
Item Listings	5,402
User Reviews + FAQs	247,258
Conversational Dataset	
User intents	297,552
Interactive Stories	5
Domain Knowledge Graph	
Nodes	247,376
Edges	418,817
Common Sense Domain Knowledge Graph	
Nodes	12,443,574
Edges	36,193,223

5.1.2 Building the Common-sense Domain Knowledge Graph

The common-sense domain knowledge graph is built as follows:

Bulk-load processed Airbnb domain graph data generated in 4.3 using cypher scripts.Below Fig. 11 is a snapshot of the knowledge graph generated at this step.

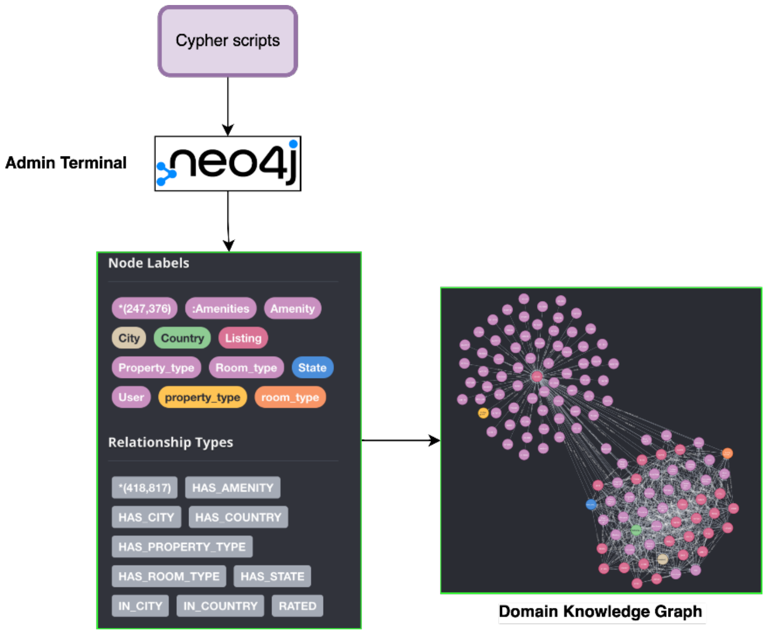


Fig. 11. Building the domain knowledge graph.

Merge the ConceptNet5(common-sense knowledge graph) [6] embeddings into thereview text and listing text. The pre-trained embeddings can be downloaded from: ConceptNet5 NumberBatch [17]. Below Fig. 12 is a snapshot of the expanded common-sense knowledge graph generated at this step.

This common-sense domain knowledge graph can be queried using cypher scripts inreal-time to generate Airbnb item recommendations. Three types of recommendation approaches are implemented independently and connected to the conversational channelfor evaluating their performance as follows:

Real-time collaborative filtering: Below Fig. 13 shows an example query and recommendations generated for user-id ' 10952' where the recommendations are ranked by Jaccard similarity computation, and top recommendations are returned asa real-time response.

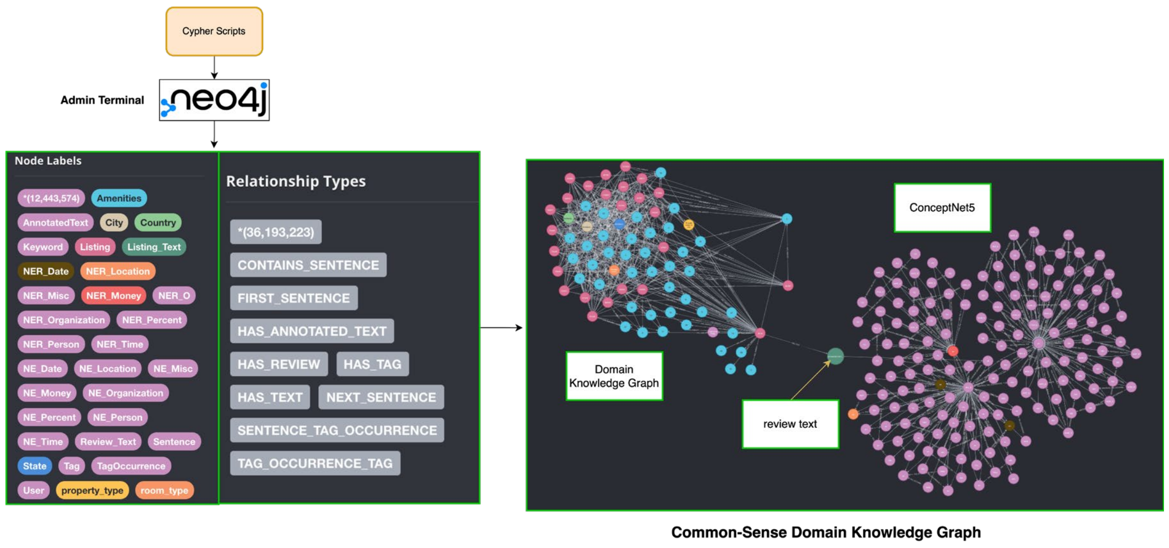
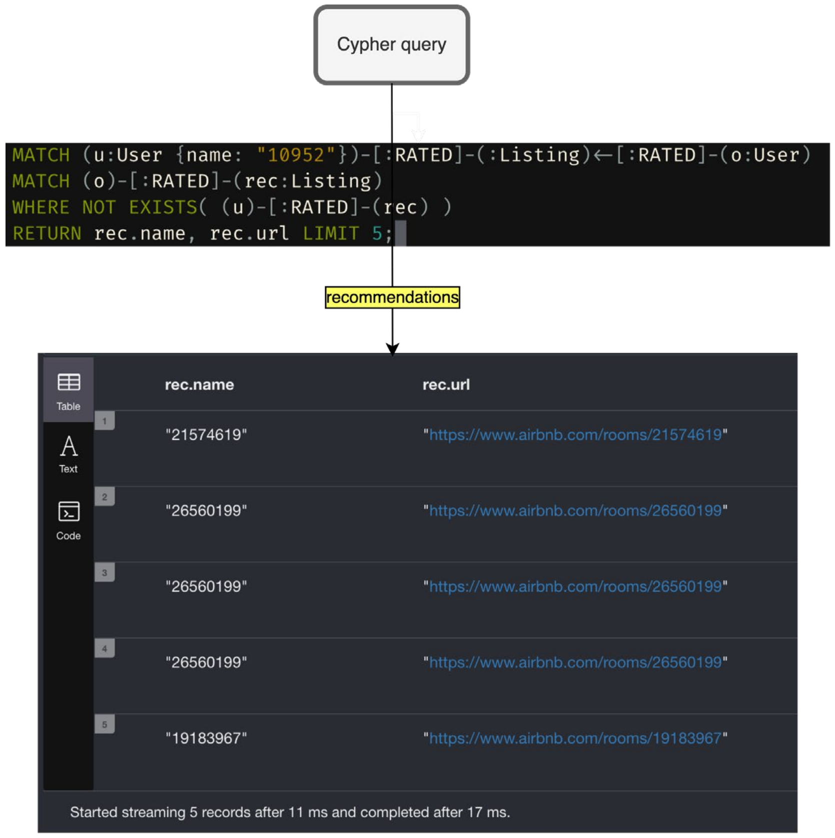


Fig. 12. Common-Sense domain knowledge graph.

Real-time content-based filtering: Below Fig. 14 shows an example query and recommendations generated for user-id ’10952’ based on the historical user ratingson the item listings filtered by amenities. The recommendations generated are rankedby jaccard similarity computation and top recommendations are returned as a real-time response.



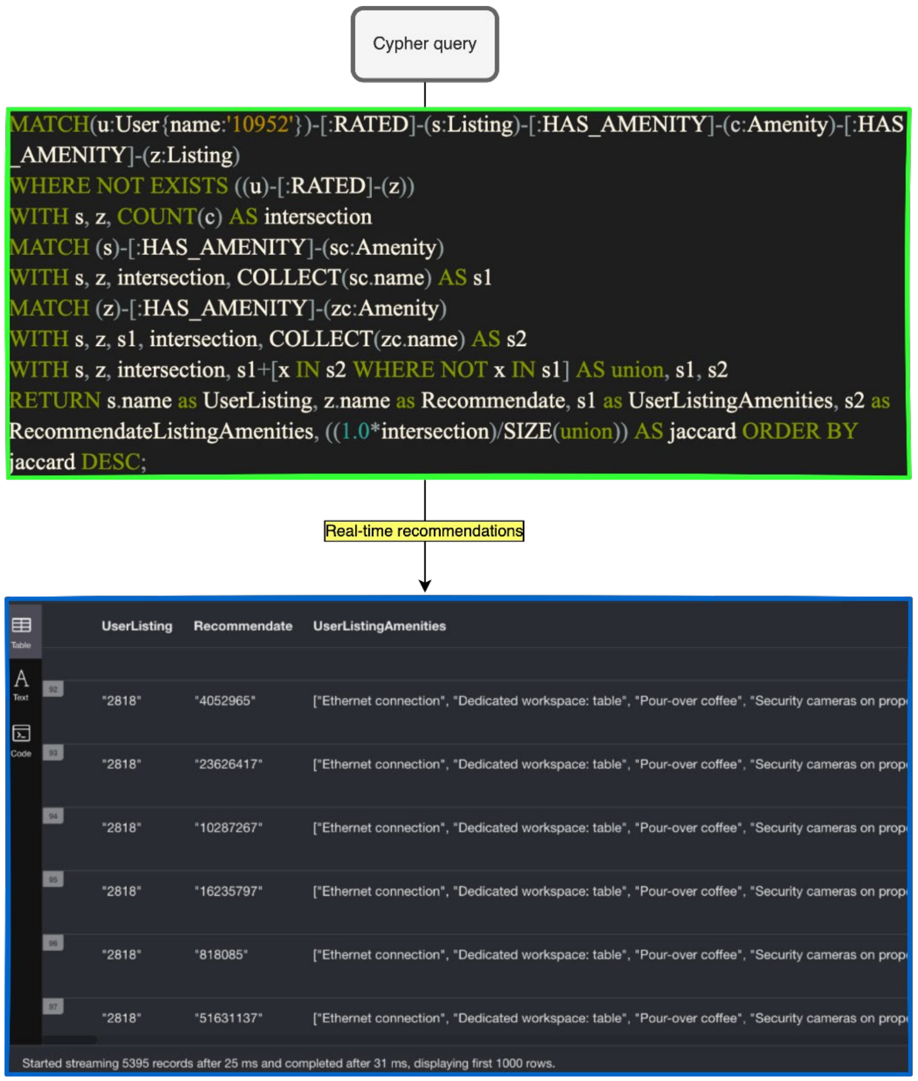
```
MATCH (u:User {name: "10952"})-[:RATED]-(:Listing)←[:RATED]-(o:User)
MATCH (o)-[:RATED]-(rec:Listing)
WHERE NOT EXISTS( (u)-[:RATED]-(rec) )
RETURN rec.name, rec.url LIMIT 5;
```

	rec.name	rec.url
1	"21574619"	"https://www.airbnb.com/rooms/21574619"
2	"26560199"	"https://www.airbnb.com/rooms/26560199"
3	"26560199"	"https://www.airbnb.com/rooms/26560199"
4	"26560199"	"https://www.airbnb.com/rooms/26560199"
5	"19183967"	"https://www.airbnb.com/rooms/19183967"

Started streaming 5 records after 11 ms and completed after 17 ms.

Fig. 13. Real-time recommendation using collaborative filtering.

Even a small number of irrelevant itemscan break the user engagement leading to unsuccessful conversations. Fig. 15 showsan example query and recommendations generated using the common-sense domainknowledge graph based on the user utterances. For example, the user asks, ’find me aplace with a refrigerator,’ the RASA NLU identifies the ’refrigerator’ entity and passes it on to the querying interface. A conceptual query is done for the entity ’refrigerator’ using the common-sense knowledge graph, which fetches all the closelyrelated concepts to the entity refrigerator like keep-food-cold, minifridge, frozen chicken, etc. Unlike traditional recommendation approaches, a conceptual matching of the item attributes ensures a no-cold start problem for new users and items. The recommendations are computed by including the top similar concepts to the entity extracted from the user utterances.



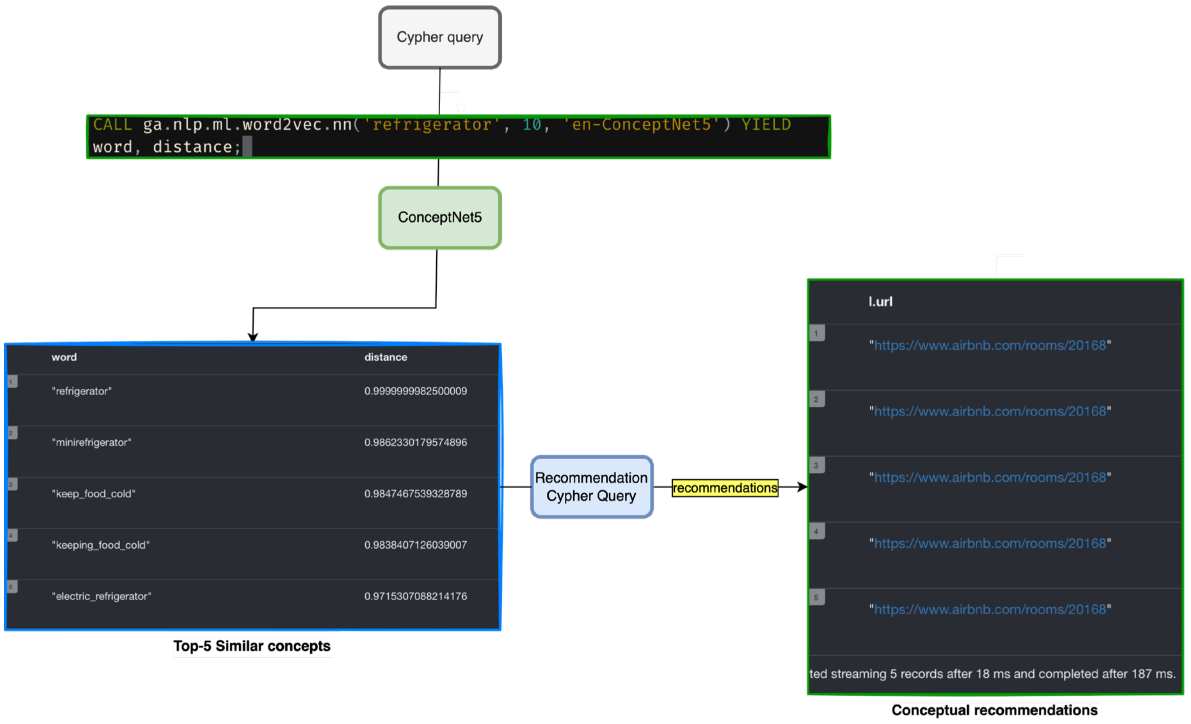
```
MATCH(u:User{name:'10952'})-[:RATED]-(s:Listing)-[:HAS_AMENITY]-(c:Amenity)-[:HAS
_AMENITY]-(z:Listing)
WHERE NOT EXISTS ((u)-[:RATED]-(z))
WITH s, z, COUNT(c) AS intersection
MATCH (s)-[:HAS_AMENITY]-(sc:Amenity)
WITH s, z, intersection, COLLECT(sc.name) AS s1
MATCH (z)-[:HAS_AMENITY]-(zc:Amenity)
WITH s, z, s1, intersection, COLLECT(zc.name) AS s2
WITH s, z, intersection, s1+[x IN s2 WHERE NOT x IN s1] AS union, s1, s2
RETURN s.name as UserListing, z.name as Recommendate, s1 as UserListingAmenities, s2 as
RecommendateListingAmenities, ((1.0*intersection)/SIZE(union)) AS jaccard ORDER BY
jaccard DESC;
```

```
UserListing   Recommendate   UserListingAmenities

92  "2818"    "4052965"     ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop
93  "2818"    "23626417"    ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop
94  "2818"    "10287267"    ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop
95  "2818"    "16235797"    ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop
96  "2818"    "818085"      ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop
97  "2818"    "51631137"    ["Ethernet connection", "Dedicated workspace: table", "Pour-over coffee", "Security cameras on prop

Started streaming 5395 records after 25 ms and completed after 31 ms, displaying first 1000 rows.
```

Fig. 14. Real-time recommendation using content-based filtering.

5.1.3 Training the Dialog Component

An NLU pipeline processes the input text to classify intents and extract entities. The conversational data was split into training and test sets in a 70:30 ratio and trained for 1000 epochs using the DIETClassifier [33]. In contrast, the Rasa Core model [28] is trained using the MemoizationPolicy and RulePolicy UnexpecTEDIntentPolicy for 100 epochs and the TED policy for 1000 epochs.



```
CALL ga.nlp.ml.word2vec.nn('refrigerator', 10, 'en-ConceptNet5') YIELD
word, distance;
```

	word	distance
1	"refrigerator"	0.9999999982500009
2	"minirefrigerator"	0.9862330179574896
3	"keep_food_cold"	0.9847467539328789
4	"keeping_food_cold"	0.9838407126039007
5	"electric_refrigerator"	0.9715307088214176

	l.url
1	"https://www.airbnb.com/rooms/20168"
2	"https://www.airbnb.com/rooms/20168"
3	"https://www.airbnb.com/rooms/20168"
4	"https://www.airbnb.com/rooms/20168"
5	"https://www.airbnb.com/rooms/20168"

```
ted streaming 5 records after 18 ms and completed after 187 ms.
```

Fig. 15. Real-time recommendation using conceptual querying.

As shown in Fig. 16 the below configurations are used for the NLU pipeline and theRASA Core policies.

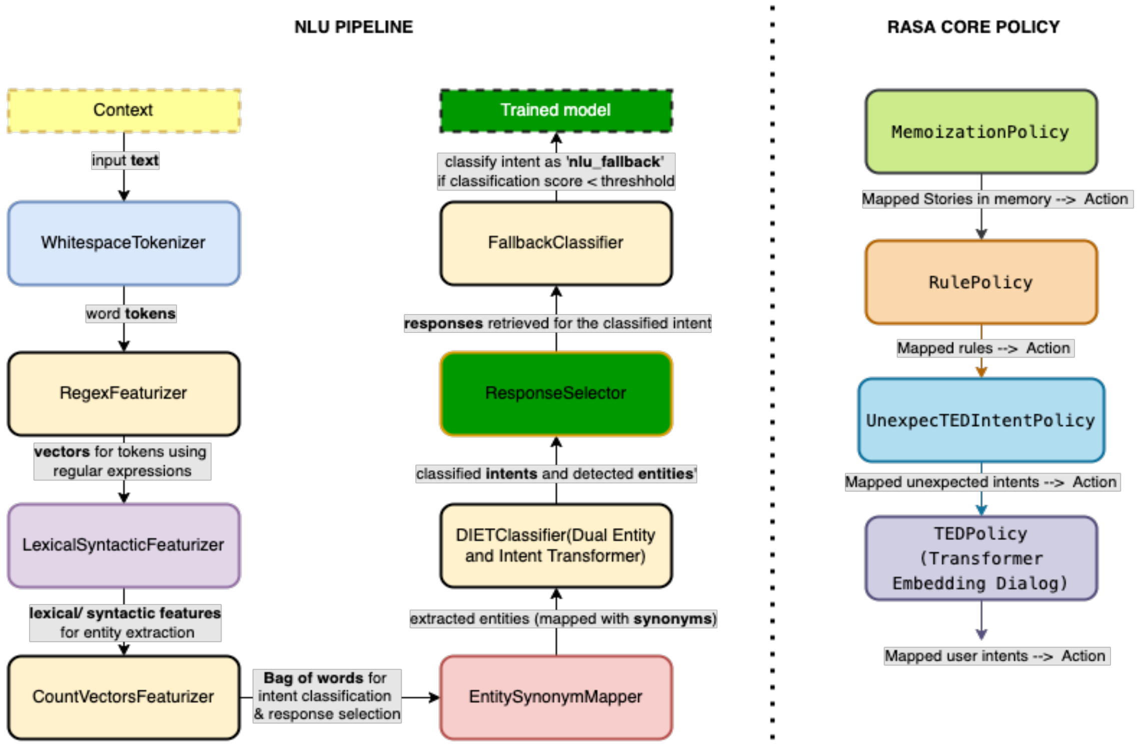

NLU PIPELINE
Context
input text
WhitespaceTokenizer
word tokens
RegexFeaturizer
vectors for tokens using regular expressions
LexicalSyntacticFeaturizer
lexical/ syntactic features for entity extraction
CountVectorsFeaturizer
Bag of words for intent classification & response selection
EntitySynonymMapper
extracted entities (mapped with synonyms)
DIETClassifier(Dual Entity and Intent Transformer)
classified intents and detected entities
ResponseSelector
responses retrieved for the classified intent
FallbackClassifier
classify intent as 'nlu_fallback' if classification score < threshhold
Trained model
RASA CORE POLICY
MemoizationPolicy
Mapped Stories in memory --> Action
RulePolicy
Mapped rules --> Action
UnexpecTEDIntentPolicy
Mapped unexpected intents --> Action
TEDPolicy (Transformer Embedding Dialog)
Mapped user intents --> Action

Fig. 16. RASA solution configuration.

Configurations for the NLU pipeline, the following configurations are used for theNLU task [27].

WhitespaceTokenizer tokenizes the input text into individual words(tokens).

RegexFeaturizer to create vectors for the individual words(tokens) using thedefined regular expressions.

LexicalSyntacticFeaturizer to create lexical and syntactic features for entityextraction.

CountVectorsFeaturizer to create a Bag-of-words representation to aid intent classification and response selection.

EntitySynonymMapper to match the extracted entities with synonymous words.

DIETClassifier(Dual Entity and Intent Transformer), a multi-task transformer-based architecture, is included to perform intent classification and entity extraction. This model uses a Conditional Random Field (CRF) layer on topof the transformer output sequence to map the corresponding input sequences.

ResponseSelector to predict the responses returned to the user for the classifiedintent.

FallbackClassifier to generate a fallback response defined in the domain configuration in case of intent classification score below a marked threshold.

5.1.4 Baselines

SMARTREC implements a real-time CRS where the conversational model and the recommendation engine have to be tied together to develop appropriate recommendationsfor the user. So, this study will evaluate both the recommendation and the conversational components separately to state the final efficiency of the system.

The following baselines are considered to evaluate the recommendation task:

KGSF: This work combines the item and the word vector spaces by adopting Mutual information maximization [11]. It further builds a recommendation model using a selfattentive gating network to rank and generate item recommendations. Finally, a transformer-based language model generates natural language responsescontaining the recommended items.

PySpark ALS model: This is a neural collaborative filtering approach [34]. A set oflatent factors represents the users and the items. The ALS algorithm trains the models to generate recommendations depending on the predicted ratings.

Gensim Doc2Vec model: This is a content-based filtering approach using a basic neural network model where document vectors are created using the word embeddings for every user review and the item description [35]. The vector representations of the given user utterance are compared to the trained vector embeddings using cosine similarity metrics to compute the closeness of the items tothe user interests and generate recommendations accordingly.

Neo4j collaborative filtering: This is a real-time user-based collaborative filtering approach where the closely matched items with the user interests are fetched from the pre-loaded semantic graph embeddings [25].

Neo4j content-based filtering: This is a real-time content-based filtering approach where the closely matched items with the user interests are fetched from the pre-loaded semantic graph embeddings [25].

The following baselines are considered to evaluate the conversational task:

Default RASA Configuration: This setup provides an NLU pipeline that createsvectors of the word embeddings to perform intent classification and entity extraction [26].

Semantic RASA Configuration: This improved setup includes pre-trained Wikipedia word embeddings to map the word vectors to a set of contextually similarwords in the vector space to classify intents and extract entities [36], [37].

The overall goal of the experiments is to analyze if the proposed approach can eliminate offline training for the recommendation task and further analyze to see if merging the common-sense ConceptNet5 knowledge graph with the domain knowledgegraph can fetch better recommendations.

5.1.5 Evaluation metrics

The experiments of this study uses the following evaluationstrategies [38]-[41]:

KGSF and SMARTREC implement CRS [42] based on the semantic merging of the common-sense knowledge graph to the item attribute to improve recommendations. This study adopts different metrics such as *'scalability', 'training time', 'dataset', 'real-time operation mode', 'interactive response', 'solution architecture', 'ConceptNet format',* and *'runtime environment'* to show how SMARTREC improvesupon KGSF.

This study compares the real-time recommendations generated using the Graph database and offline recommendations retrieved from the vector models using training time, response time and *'explainability'* metrics [43] listed below to measurethe relevance of recommendations to the user expectations.

Exact (E): recommended item is most relevant to the user intent. (e.g., *'kettle'* fora *'hot water kettle' query.*)

Substitute (S): recommended item is partially relevant to the user intent. (e.g., *'bathtub'* for a *'pool' query.*)

Complement (C): The recommended item does not exactly match the user intentbut could be related conceptually to the query. (e.g., *'range with oven'* for a *'dishwasher' query.*)

Irrelevant (I): recommend items that do not match the user intent. (e.g., *'babycrib'* for a *'parking' query.*)

This study compares the conversational task using different metrics such as *’trainingtime’*, *’intent classification score’, ’entity extraction score’*.

5.2 Evaluation on the CRS

This study compares the effectiveness of the proposed model with the semantic CRS approach called KGSF using different parameters shown in Table 2.

Table 2
Compartive Study – KGSF and SMARTREC

Metric	SMARTREC	KGSF
Scalability	Highly scalable graph database	File-based data
Training time	23.68 minutes	33 hours
Real-time	Yes	No
Interactive response	Yes	No
Architecture	Distributed	Stand-alone
ConceptNet format	Pre-trained Vector Embeddings	Create Vector Embeddings
Runtime environment	No GPU required	GPU required
	DATASET	
	Conversational Dataset	
Name	Self-curated	DIALog (REDIAL)
No. of utternaces (user intents)	310,812	182,150
	Item Dataset	
Name	5,407	51,699
Item reviews	247,258	No historical interactions used

For the recommendation engine, SMARTREC improves upon KGSF by using a highly scalable distributed graph database to represent the knowledge graphs, whereas KGSF uses file-based stand-alone knowledge graphs. Compared to KGSF, SMARTRECdrastically reduces the training time to create the graph embeddings from approximately33 hours to only 23.68 minutes for the common-sense domain knowledge graph.

SMARTREC implements a real-time recommendation engine using the graph database, whereas KGSF is an offline traditional recommendation approach. SMARTREC providesan interactive response for the conversational task, whereas KGSF provides a generated response. SMARTREC removes the need for intensive training on the machine learning models using GPUs to execute the deep learning experiments, whereas KGSF experimentspresent a GPU requirement. Furthermore, SMARTREC merges the common-sense knowledge graph called ConceptNet5 into the domain knowledge graph as pre-computedword embeddings in a few minutes. In contrast, KGSF merges the ConceptNet3 data using Mutual Information Maximization, a computationally intensive operation.

For the conversational task, SMARTREC uses a self-curated dataset using thehistorical user reviews for the itemset. In contrast, KGSF uses a crowd-sourced DBPedia [18] conversational dataset to represent user intents. By doing this, SMARTRECgrasps the users' intent for a particular domain, eliminates potential bias, and generalizeswell on the resulting language models.

Overall, SMARTREC provides a much more accurate semantic representation of the domain knowledge, similar to how human brains process information and render real-time explainable [43] recommendations using graph traversal algorithms [44].

5.3 Evaluation on the Recommendation framework

5.3.1 Quantitative analysis

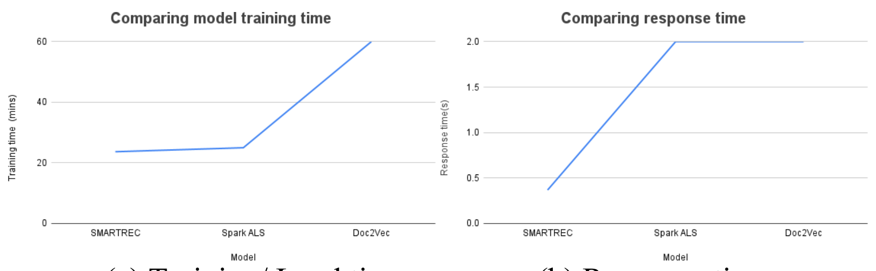
As shown in Table 3 and Table 4, this study conducts extensive experiments to compare the MAR@K, training time and response time for traditional recommendation approaches [8], [9] against the SMARTREC real-time recommendation system [45]. Asa result, SMARTREC outperforms the traditional approaches by a large margin in termsof MAR@K, training time, and good performance in terms of real-time response time, asshown in Fig. 17a, Fig. 17b, Fig. 18a and Fig. 18b which is a significant improvement fora real-time CRS, including a substantial common-sense knowledge graph like ConceptNet5. Finally, SMARTREC implements a real-time recommendation frameworkwith a much faster training and response time, eliminating offline training using traditional recommendation approaches. MAR@K for recommendation approaches, are logged for reference.

Table 3
Evaluation – Recommendation Task Performances. Results of different expierments for the recommendation tasks are recorded. Training time in minutes-*lower is better*, Response time in seconds *– lower is better*, ESCIN – *(Except to No Results) - (Higher to lower is better)*

Quantitative Analysis				
Task	**Method**	**Mode**	**Training time (min)**	**Response time(s)**
Content-based filtering	Doc2Vec	Offline	25	2
Content-based filtering	Cypher query	Real-time	0.13	0.0015
Colaborative filtering	Cypher query	Offline	0.13	0.0015
Colaborative filtering	Spark ALS	Offline	25	2
DKG	Cypher query	Real-time	0.13	0.368
CS-DKG	Cypher query	Real-time	23.68	0.368
Qualitative Analysis				
DKG				
Expect (%)	**Substitute (%)**	**Complement (%)**	**Irrelevant (%)**	**No results (%)**
30	2.5	5	12.5	50
CS-DKG				
Expect (%)	**Substitute (%)**	**Complement (%)**	**Irrelevant (%)**	**No results (%)**
62.5	20	10	0	7.5

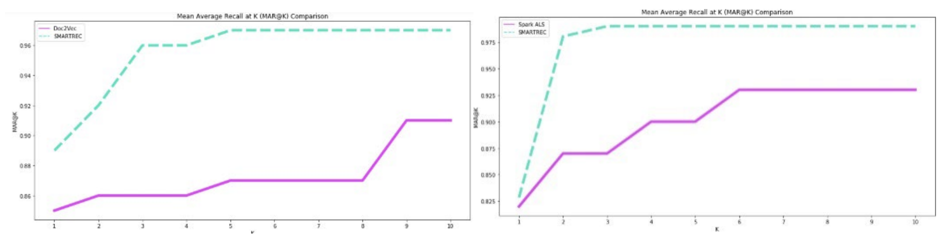
Table 4
MAR@K for different recommendation approaches. Results of MAR@K (0-1) – *higher is better*

Quantitative Analysis				
MAR@K	Spark ALS	SMARTREC CF	Doc2Vec	SMARTREC CBF
K=1	0.85	0.83	0.85	0.89
K=2	0.86	0.98	0.86	0.92
K=3	0.86	0.99	0.86	0.96
K=4	0.86	0.99	0.86	0.97
K=5	0.87	0.99	0.87	0.97
K=6	0.87	0.99	0.87	0.97
K=7	0.87	0.99	0.87	0.97
K=8	0.87	0.99	0.87	0.97
K=9	0.91	0.99	0.91	0.97
K=10	0.91	0.99	0.91	0.97



(a) Training/ Load time (b) Response time

Fig. 17. Quantitative analysis - Training and Response times for different recommendation approaches.



(a) MAR@K Collaborative filtering (b) MAR@K Content-based filtering
Fig. 18. Quantitative analysis - MAR@K for different recommendation approaches.

5.3.2 Qualitative analysis

As shown in Table 3, this study conducted a real-time conceptual query on the domain knowledge graph and the common-sense domain knowledge graph [40], based upon the item attribute searched and evaluated the top-1 recommended item for a batch of user queries using the defined metrics such *Exact*, *Substitute*, *Complement*, *Irrelevant*, *No results* and comparative study presented in Fig. 19. Evaluation using ESCIN metrics, arelogged for reference.

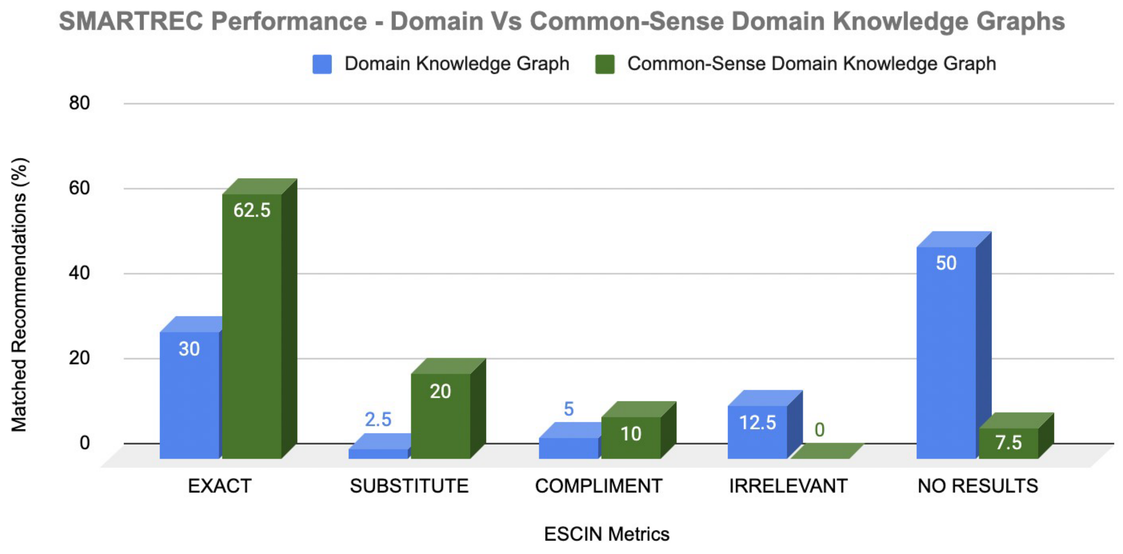


Fig. 19. ESCIN— Performance - Domain knowledge Graph Vs Common-SenseKnowledge Graph.

5.4 Evaluation on the Conversational framework

As shown in Table 5 below, this study evaluates the performance of the language models using the default RASA pipeline against the semantically improved pipeline usingWikipedia embeddings. The models using pre-trained Wikipedia semantic embeddings showed significant improvement as shown in Fig. 20a, Fig. 20b and Fig. 20c in terms ofF1, Recall, Precision and training time both for the intent classification score and the entityextraction. Score [41].

Table 5
Conversational Task Performances. Results of different experiments for the conversational tasks are recoded. Training time in minutes – *lower is better* and Score (0-1) – *higher is better*

Model Training

Task	**Method**	**Mode**	**Dataset**	**Time(hour)**
NLU, RASA Core	Default Configuration	Offline	Conversational Data	1.50
NLU, RASA Core	Semantic Configuration	Offline	Conversational Data	0.83

Model Performance

Metrics	Default pipeline(Score)	Semantic pipeline(Score)
Intent classification score		
F1	0.82	0.99
Recall	0.70	0.99
Precision	0.98	1
Entity extraction score		
F1	0.86	0.99
Recall	0.80	0.99
Precision	0.95	1

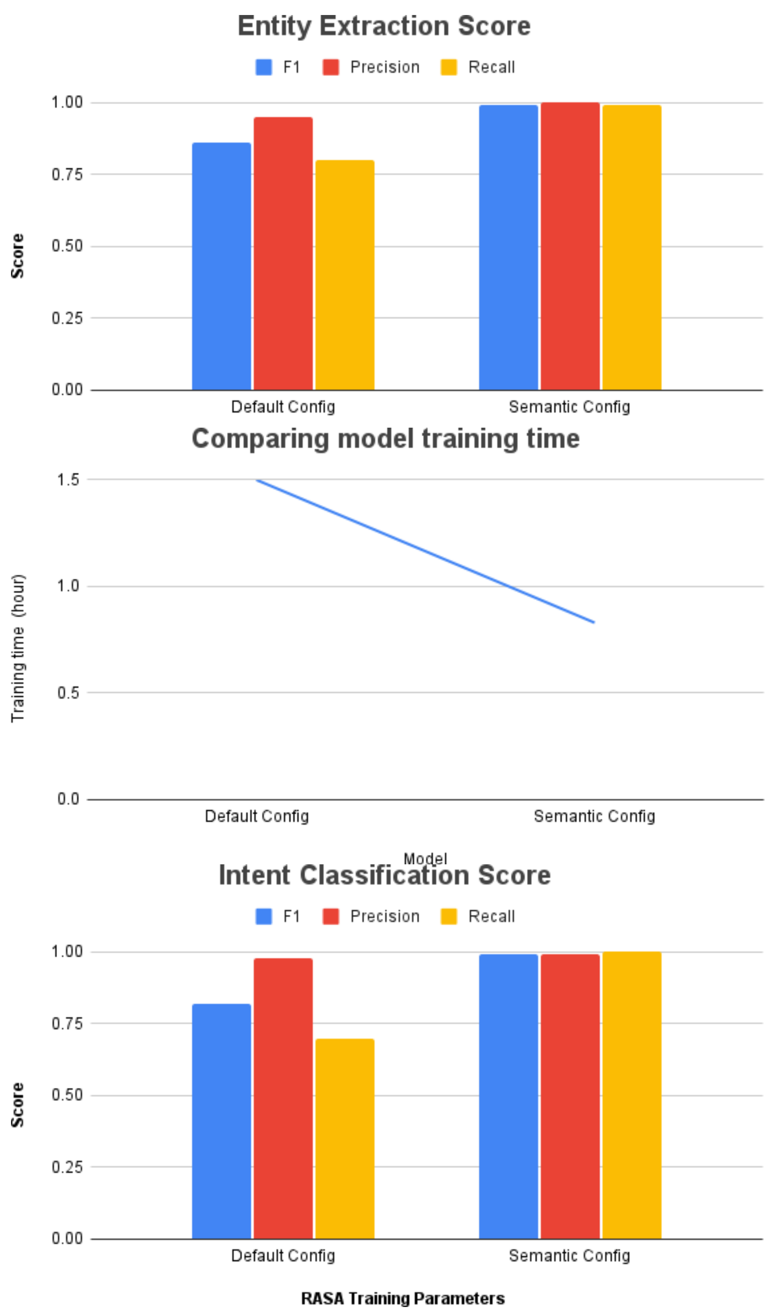


Fig. 20. Quantitative analysis for different Conversational AI approaches.

6 FUTURE WORK

This study will consider including multiple intents in the user requirements and moreitem listings from global locations in future work. This study will also consider evaluatingthe solution on different real-time database engines like TigerGraph, and ArangoDB to improve scalability and reduce query response time from seconds to milliseconds.

Though this study comes up with conceptual item listing recommendations by successfully understanding the user requirements in an interactive conversation, the testing scope for this project is limited. Therefore, this study will further consider adapting A/B testing [46] strategies with metrics such as CTR, Search CVR, etc., to evaluate the real-time recommendation system through user interactions. Furthermore, this study will consider implementing a multilingual common-sense knowledge graph in the future. In addition, this study will also consider including the human evaluation and defineadditional metrics such as fluency and informativeness to evaluate the CRS performances.

In future work, this study will consider incorporating an image, voice-based interactions for the conversation task, and multilingual CRS to support global audiences.

7 CONCLUSIONS

This paper proposes a unique real-time Knowledge Graph-based semantically intelligent interactive conversational recommendation engine to recommend top matcheditems to the user. The proposed system will leverage the RASA development platform anda graph-based recommendation engine using Neo4j. The common-sense knowledge graphenhanced recommendation and querying interface created using the ConceptNet5 numberbatch embeddings offers relevant search, outperforms the vector-based recommendation frameworks' performance, and generates context-aware recommendations. Furthermore, the semantic understanding is included in the NLU model as a pre-trained common sense vector eliminating the need to train from scratch. This approach will save the training time required to combine the item and common-sensedata into one unified semantic dataset. Additionally, this study will incorporate only relevant historical user reviews on the itemset and the item entities to conduct the conversation with a predefined user persona with domain-relevant auto-generated mock conversational data instead of unspecific crowd-sourced conversational data.

The work will be adaptable to many online platforms, including big players like Airbnb, Amazon, Netflix, eBay, Wish, Zocdoc, etc. The system will help the e-commerceplayers improve customer experience and provide an AI-enabled customer recommendation and support system. This study focuses on an Airbnb-inspired use caseusing the large amount of open-source global data provided by Airbnb. Airbnb has implemented a task-oriented customer support bot to provide customer support during theCOVID pandemic and has added flexible filters to tackle the rigidity of the current product search system. SMARTREC can significantly expand these capabilities with a more sophisticated AI-based conversational recommendation approach.

Literature Cited

[1] A. Singh. “Harnessing conversational voice AI in the e-commerce industry.” https://www.forbes.com/sites/forbestechcouncil/2021/08/24/harnessing-conversational-voice-ai-in-the-e-commerce-industry/?sh=30d9bee760ec (accessed March 21, 2022).

[2] Y. Zhang, X. Chen, Q. Ai, L. Yang, and W. B. Croft, “Toward conversational search and reccomendation: System ask, user respond,” in *Proceedings 27th acm International Conference Information and Knowledge Management*, pp. 177-186, 2018.

[3] W. L. Woo, B. Gao, R. R. O. Al-Nima, and W.-K. Ling, “Development of conversational artificial intelligence for pandemic healthcare query support,” *International Journal of Automation, Artificial Intelligence and Machine Learning*, vol. 1, no. 1, pp. 54–79, 2020.

[4] T. Zhang, Y. Liu, P. Zhong, C. Zhang, H. Wang, and C. Miao, “Kecrs: Towards knowledge-enriched conversational recommendation system,” *arXiv preprint arXiv:2105.08261*, 2021.

[5] “Get the Data.” http://insideairbnb.com/get-the-data.html (accessed March 21, 2022).

[6] R. Speer, J. Chin, and C. Havasi, “Conceptnet 5.5: An open multilingual graph of general knowledge,” in *Thirty-first AAAI conference artificial intelligence*, 2017.

[7] D. Jannach, A. Manzoor, W. Cai, and L. Chen, “A survey on conversational recommender systems,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–36, 2021.

[8] D. Das, L. Sahoo, and S. Datta, “A survey on recommendation system,” *International Journal of Computer Applications*, vol. 160, no. 7, pp. 6-10, 2017.

[9] M. Sridevi, R. R. Rao, and M. V. Rao, “A survey on recommender system,” *International Journal of Computer Science and Information Security*, vol. 14, no. 5,p. 265, 2016.

[10] K. Bradley and B. Smyth, “Improving recommendation diversity,” in *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science, Maynooth, Ireland*, vol. 85, pp. 141–152, 2001.

[11] K. Zhou, W. X. Zhao, S. Bian, Y. Zhou, J.-R. Wen, and J. Yu, “Improving

conversational recommender systems via knowledge graph based semantic fusion," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1006–1014, 2020.

[12] Y. Zhang, *et al.*, "Dialogpt: Large-scale generative pre-training for conversational responsegeneration," *arXiv preprint arXiv:1911.00536*, 2019.

[13] C.-M. Wong, *et al.*, "Improving conversational recommender system by pretraining billion-scale knowledge graph," in *2021 IEEE 37th InternationalConference on Data Engineering (ICDE)*, pp. 2607–2612, 2021.

[14] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*,pp. 173–182, 2017.

[15] J. Gao, M. Galley, and L. Li, "Neural approaches to conversational ai," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 1371–1374, 2018.

[16] Y. Liu, S. Wang, M. S. Khan, and J. He, "A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering," *Big Data Mining and Analytics*, vol. 1, no. 3, pp. 211–221, 2018.

[17] N. Van Hoang, S. H. Mulvad, D. N. Y. Rong, and Y. Yue, "Zero-shot learning in named-entity recgonition with external knowledge," *arXiv preprint arXiv:2111.07734*, 2021.

[18] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The Semantic Web*, pp. 722–735, 2007.

[19] S. B. Ahire and H. K. Khanuja, "A personalized framework for health care recommendation," in *2015 International Conference on Computing Communication Control and Automation*, pp. 442–445, IEEE, 2015.

[20] T. Bocklisch, J. Faulkner, N. Pawlowski, and A. Nichol, "Rasa: Open source language understanding and dialogue management," *arXiv preprint arXiv:1712.05181*, 2017.

[21] R. Van Bruggen, *Learning Neo4j*. Birmingham, UK, Packt Publishing Ltd, 2014.

[22] "Help Center." https://www.airbnb.com/help/contact-us, 2021 (accessed March 21, 2022).

[23] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60, 2014.

[24] "Questgen AI." https://github.com/ramsrigouthamg/Questgen.ai (accessed March 21, 2022).

[25] "Tutorial: Build a Cypher Recommendation Engine." https://neo4j.com/developer/cypher/guide-build-a-recommendation-engine/ (accessed March 21, 2022).

[26] M. Hahn and M. Baroni, "Tabula nearly rasa: Probing the linguistic knowledge of character-level neural language models trained on unsegmented text," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 467–484, 2019.

[27] "NLU Training Data." https://rasa.com/docs/rasa/nlu-training-data/ (accessed March 21, 2022).

[28] "Rasa.core.processor." https://rasa.com/docs/rasa/reference/rasa/core/processor/ (accessed March 21, 2022).

[29] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[30] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.

[31] E. McAuliffe, "Enterprise Knowledge Graph Trends for." https://www.tigergraph.com/blog/enterprise-knowledge-graph-trends-for-2021/ (accessed March 21, 2022).

[32] "Building & exploring a music knowledge graph." https://graphaware.com/blog/engineering/building-and-exploring-music-knowledge-graph.html (accessed March 21, 2022).

[33] J. Hao, R. Li, H. Fei, and Y. Tao, “A medical dialogue system based on diet and knowledge graph,” in *2021 International Conference on Health Big Data and Smart Sports (HBDSS)*, pp. 56–59, IEEE, 2021.

[34] B. Kıran Çelebi, “Product recommendation for c2c marketplace with collaborative filtering als algorithm,” 2021.

[35] R. Řehřek, P. Sojka, *et al.*, “Gensim—statistical semantics in python,” Genism.org, 2011.

[36] “Exploring Semantic Map Embeddings / Part I.” https://rasa.com/blog/exploring-semantic-map-embeddings-1/ (accessed March 21, 2022).

[37] “Exploring Semantic Map Embeddings / Part II.” https://rasa.com/blog/exploring-semantic-map-embeddings-2/ (accessed March 21, 2022).

[38] O. Alkan, E. M. Daly, and A. Botea, “An evaluation framework for interactive recommender systems,” in *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, pp. 217–218, 2019.

[39] Y. Zhao, C. Wang, H. Han, M. Shu, and W. Wang, “An impact evaluation framework of personalized news aggregation and recommendation systems,” in *2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, pp. 893-900, IEEE, 2020.

[40] “Amazon KDD Cup’ 22.” https://www.aicrowd.com/challenges/esci-challenge-for-improving-product-search (accessed March 21, 2022).

[41] “Testing Your Assistant.” https://rasa.com/docs/rasa/testing-your-assistant/ (accessed March 21, 2022).

[42] W. Lei, X. He, M. de Rijke, and T.-S. Chua, “Conversational recommendation: Formulation, methods, and evaluation,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2425–2428, 2020.

[43] M. Palmonari and P. Minervini, “Knowledge graph embeddings and explainable ai,” *Knowledge Graphs for Explainable Artificial Intelligence: Foundations, Applications and Challenges*, vol. 47, p. 49, 2020.

[44] M. Needham and A. E. Hodler, *Graph algorithms: Practical examples in Apache Spark and Neo4j*. Seabastopol, CA, USA, O'Reilly Media, 2019.

[45] A. Anthony, Y.-K. Shih, R. Jin, and Y. Xiang, "Leveraging a graph-powered, real-time recommendation engine to create rapid business value," in *Proceedings ofthe 10th ACM Conference on Recommender Systems*, pp. 385–386, 2016.

[46] D. Siroker and P. Koomen, *A.B testing: The most powerful way to turn clicks into customers.* New York, NY, USA, John Wiley & Sons, 2013.