GUApp: a Knowledge-aware Conversational Agent for Job Recommendation

Giovanni Maria Biancofiore[1,2], Tommaso Di Noia[1], Eugenio Di Sciascio[1], Fedelucio Narducci[1] and Paolo Pastore[1,2]

[1]Polytechnic University of Bari, Bari, Italy
[2]Corresponding authors

Abstract

GUapp is an ecosystem for job-postings search and recommendation for the Italian public administration. Its main goal is to match user skills and requests with job positions available on the Gazzetta Ufficiale website, offering recommendation services in a conversational setting. Guapp's dialogues are modelled employing a domain-specific Knowledge Graph, which improves the users' natural language interaction with the app as well as the user experience. Thanks to that, the search and recommendation process becomes incremental and the user can dynamically provide her preferences at each stage of the interaction. In this paper, we present GUapp and its overall architecture, besides the functioning of the conversational agent that dialogues with the user by exploiting a custom-designed Knowledge Graph. We also show a running example that outline how GUapp models users and provides them effective recommendations through natural language conversations.

Keywords

Recommender System, Public Administration, Knowledge Graph, Conversational Recommender System

1. Introduction

The information overload is a well-known problem that impacts the digital experience of users when they need to find interesting items in a large set of possible options [1]. That is the case of looking for a book to read, a smartphone to buy, a TV series to watch, and so on. Users especially perceived the same issue when they are looking for a new job, where the only available strategy is to search for interesting job calls. Moreover, job calls have a limited period for applying. For this reason, it is crucial to constantly look for attractive job openings. In this scenario, a system that only allows to search by a query composed of a set of relevant keywords can make this task an ordeal for users.

The GUapp platform has been designed and developed to find and discover job positions among job offers in the Italian public administration[1]. This problem has been investigated in the literature from two different perspectives: Information Retrieval and Information Filtering. From the Information Retrieval side, there are systems that generally do not take into account the user's past preferences, and only retrieve the most relevant documents based on the current user query. Recommender Systems (RS) are Information Filtering tools for suggesting services and items tailored to the specific users' characteristics and requests. In our case, the list of job calls is daily updated and provided to the user depending on her past preferences.

GUapp offers a natural-language based interaction through a chatbot, which allows the user to define her interests, describe her skills, and filter out results that did not match with her requirements. Moreover, our system leverage the felt issues of cold-start and the possible lack of items to suggest with a Conversational Agent which interacts with users exploiting a domain-specific Knowledge Graph (KG), differently from the previous version of the tool [2]. The GUapp KG is obtained by merging some sub-graphs from state-of-the-art solutions like Dbpedia[2] and new triples generated from data scraped from external sources such as the ISTAT[3] website. From the latter, we have taken information about profession hierarchies and fields to which jobs belong. Furthermore, we have built an ontology on which the retrieved facts rely. This KG allows our system to search for new semantically linked user preferences, besides engaging a negotiation phase when the proposed calls do not match all the user requirements. Exploiting the KG relations, GUapp can search for jobs that do not perfectly suit the user preferences but still remaining close to her interests. On this line, conversations can reach a finer grained level of de-

3rd Edition of Knowledge-aware and Conversational Recommender Systems (KaRS) & 5th Edition of Recommendation in Complex Environments (ComplexRec) Joint Workshop @ RecSys 2021, September 27–1 October 2021, Amsterdam, Netherlands
giovannimaria.biancofiore@poliba.it (G. M. Biancofiore);
tommaso.dinoia@poliba.it (T. Di Noia);
eugenio.disciascio@poliba.it (E. Di Sciascio);
fedelucio.narducci@poliba.it (F. Narducci);
paolo.pastore1@poliba.it (P. Pastore)

[1]All the documents are freely available online at https://www.gazzettaufficiale.it/30giorni/concorsi

[2]https://www.dbpedia.org/
[3]https://www.istat.it/

tails on job aspects to recommend to users and enhance its expressiveness as well.

2. Related Work

The proposed work is between two principal research areas: recommender systems and conversational agents. Recommender Systems (RSs) support the user during the decision making process when she has to decide among a large set of different options. RSs are grouped into two main categories: Collaborative Filtering (CF) and Content-based (CB). The CF exploits the user community in order to identify items potentially interesting for a given individual, following the intuition that similar users like similar items. Conversely, CB systems try to suggest items matching user preferences with items descriptions [1]. GUapp implements a CBRS. Actually, the main goal of the RS behind GUapp is to match the user requests/preferences with the textual description of the job proposal. A lot of studies investigated the specific task of job recommendation [3]. In the past, the most used approaches for job recommendation were based on boolean search and filtering [4]. Later, the attention has been focused on the problem of catching the user preferences and building a user profile. In [5] the authors propose a system that builds the user profile by passively detecting click-stream and read-time behaviour of users. Malinowski et al. [4] proposes a strategy based on multi-slot user profile in which different information are stored: demographic data, job experiences, languages, and IT skills. Similarly, in GUapp tool this kind of information is acquired by mean of a KG-driven conversation. Recommending items by means of a set of rules which verify whether the user tastes are satisfied or not was dealt at first by the Knowledge based Recommender Systems [6]. These systems perform reasoning on ontologies and Knowledge Bases (KBs) to find items that match the user preferences. For instance, Carrer-Neto et al. [7] opted that movies belonging to the same ontological classes of the items that the user liked in the past has to be recommended. Differently, Tarus et al. [8] exploit both ontological features and collaborative filtering to provide recommendations in the field of learning resources for some learner targets.

On the other hand, Conversational Agents are software agents that use natural language to interact with the user. They can be classified in two main classes: end-to-end and modular systems [9]. The former typically exploits Deep Learning techniques for learning a dialog model from a set of past conversations [10, 11, 12, 13, 14]. Modular systems adopt a pipeline-based agent which is composed of a set of modules, each with a specific function [15, 16, 17, 18]. The main difference between a Conversational Recommender System (CoRS) and a traditional RS is the interaction with the user that is more efficient and natural [19]. Accordingly, a CoRS let the system build the user profile during the interaction, allowing her to express preferences, by a human-like dialog. The aforementioned task is well suited to Knowledge-based and KG-based RSs. An example can be found in [20], where a comprehensive KG is built upon a specific domain to lead dialogues and recommendations. CoRS have proved to be very effective both from the recommendation and Human-Computer Interaction perspective and they have been used in different domains [21, 22]. However, to the best of our knowledge, the GUapp system represents the first attempt of implementing a CoRS for the job-recommendation task.

3. The GUapp's Architecture

In Figure 1 GUapp's architecture is sketched, composed of six main components: the Orchestrator, the Chatbot, the Recommender System, the User Profiler, the Crawler and the Knowledge Graph.

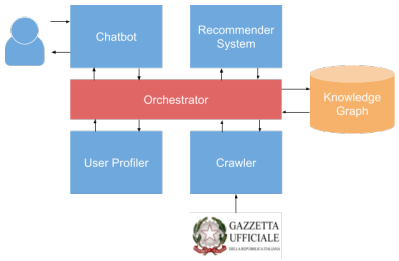

Figure 1: The GUapp's Architecture and its flow of data.

The *Orchestrator* manages the interaction between the different components of the system. For example, it invokes the RS when user asks for receiving a list of job positions based on the information stored in her profile. In detail, this module leverages the overall flow of data selecting the appropriate component to solve specific tasks. We will consider the case on which the Recommender System needs to start a negotiation phase with the user. The Orchestrator will collect data semantically related to her profile by querying the Knowledge Graph, that will be exploited to find other items interesting to the user.

The *Chatbot* is the component of GUapp which allows users to interact through natural language. It is implemented by using DialogFlow[4], a Google platform for designing and integrating conversational user interfaces, and it mainly leads conversations with users. For this purpose, the chatbot is equipped with an Intent Recognizer and an Entity Recognizer, allowing the system to understand several user requests and retrieve from her essential data for the recommending task. That is a crucial step in order to identify the back-end services to be

[4]https://dialogflow.com/

invoked for accomplishing the request. The*Intent Recognizer* analyzes the natural language request searching for specific goals such as collecting the user preferences, providing new job recommendations, or negotiating with the user. The *Entity Recognizer* is invoked in order to check whether the sentence contains mentions to real-world entities. It is implemented through Dialogflow as well, and it is powered by the KG entities. It adopts a fuzzy-matching strategy in order to identify real-world entities in the user sentence. In the case the match succeeds, the recognized entity is returned.

The *User Profiler* instead collects all the users preferences. In detail, we store all the data that they provide during conversations like favourites job locations, professions etc. When the user signs into the app for the first time, her profile is empty (i.e. *cold-start* situation), but thanks to the chatbot the user talks with GUapp about her skills, wishes, and ambitions, then the system is able to rank job calls by exploiting the provided information. The User Profile is actively updated to the new user inputs, guaranteeing the recommendations to be adaptive.

The *Recommender System* is another core component of GUapp. It exploits an Elasticsearch[5] index, which stores all the information scraped by the *Crawler* enriched with all the linked entities of our KG. In particular, for each job call, we automatically search for mentions of the KG entities or the ontological categories. The Elasticsearch documents will store the entity/category label related to each discovered mention with a confidence score, estimated with the BM25 algorithm, which shows how much the labelling process outcomes are reliable. The recommendations will be the job call closest to the user preferences with the highest confidence score.

The main intuition behind this model is the possibility to make dialogues as interactive and efficient as possible, allowing the system to negotiate with users whether results do not completely match their preferences.

To be updated with all new jobs that the market offers, GUapp implements a *Crawler* that daily extracts the job positions from *Gazzetta Ufficiale*, the official journal of record of the Italian government. It directly communicates with the Orchestrator for storing all the obtained data into the Elasticsearch instance as new documents.

Finally, we have provided to the system a domain-specific *Knowledge Graph* built upon several sources, like Dbpedia and the ISTAT website, as stated before. The latter identifies a collection of raw data that are not in KG form. As a consequence, we have modelled a new *Ontology* on which the overall KG can rely. It defines relations and hierarchies about professions, domain of competences, locations and so forth and it wisely integrates the raw information with the already structured ones. This allows GUapp to be highly knowledgeable about job features that are crucial for the recommendation task, besides granting the chatbot to leverage fine-grained conversations and negotiations with users. To make the system more specialized on the job-opening domain, we plan to enrich our KG of further facts related to the job calls an positions recommendation.

4. Building the Knowledge Sources

The more detailed a collection of data about features of items is, the higher is the accuracy of the recommendations provided by the system. Following this intuition and given the conversational configuration on which GUapp is built, we found owning a well-structured source of information an essential requisite. The system can provide more fine-grained recommendations by using a knowledge source that defines aspects users evaluate to match their interests. For instance, job location and profession that a person could cover represent two main features that people consider while seeking a new job. At the same time, skills and experiences are crucial information to retrieve the most proper job position for the user. The employment of a Knowledge Graph further allows the system to build semantically explicit user profiles. That is results in highly interpretable recommendations, besides leading efficient negotiations in case there is no item that satisfies all the user requirements.

On this line, we opted to enrich GUapp with a collection of Linked Open Data (LOD), suitable for the job recommendation task, since their availability in structured non-proprietary formats under an open license. Unfortunately, there are no KG and Ontologies already available which outline the hierarchies of job professions and their belonging fields. Accordingly, we have started to implement a new LOD resource that perfectly fits the GUapp intents, besides being also available for other related purposes. To the best of our knowledge, the GUapp KG identifies the first attempt of structuring relations between professions and their application fields under the guidelines of the LOD protocols. Moreover, it integrates also all the data related to the job recommending task obtained from other state-of-the-art solutions, like cities, regions, and countries provided by Dbpedia.

We first collected all the RDF statements about locations from the Dbpedia project and the associated ontology to create a KG that was complete and consistent for this work. For this purpose, we have exploited the OpenLink Virtuoso[6], a Dbpedia SPARQL endpoint that allowed us to perform different SPARQL queries to collect the interested data. Regarding professions and application fields, we opted to create a new ontology from scratch, assembling this information from highly reliable

[5]https://www.elasticsearch.com

[6]https://dbpedia.org/sparql/

sources. We found that the ISTAT website, managed by an Italian research institute for statistics, totally accomplish this requirement. It stored a complete hierarchy of professions organized for sectors, application fields, and services in a tree data structure navigable through web pages for each position. For example, at the higher level, we can found distinctions between intellectual, technical, and office jobs while descending into the graph groups like scientific, health, and managing positions are outlined. This taxonomy reflects what the GUapp ontology asserts about professions. The leaves of the ISTAT tree describe all the positions currently recognized in our society, like computer scientists and computer engineers, which compose some of the GUapp KG facts.

All this data are automatically retrieved from the previously mentioned website exploiting the Crawler routines. They not only collect all the job calls of the day, but they also scrape all the information that populates the KG. Then, an owl file is generated for the GUapp ontology and all the RDF triples are realized to form the KG. For instance, the Computer Engineer profile belongs to the GUapp class Electronic Engineer, a subclass of Engineering and of the higher Intellectual and highly specialized Scientific profession class. These facts are finally integrated with those obtained from Dbpedia and form the overall GUapp KG. At the current state, our KG is limited to the Italian language and it has data restricted to professions, fields, and locations. Nevertheless, thanks to its semantic structure, we plan to expand it by adding several languages besides including other job position features like user skills and job goals.

Our system relies on this knowledge source mainly for implementing two functionalities. The first one is the labelling phase of the crawled dataset that makes the recommendation possible. In detail, the job calls retrieved by the Crawler are in the form of unstructured text, so we found it necessary to index the documents on a search engine like Elasticsearch. Looking for the KG labels in each job call, we enriched all the collected texts with the GUapp linked entities, which is helpful in performing recommendations given the users' preferences. Instead, the second functionality allows the system to perform the preference elicitation and the negotiation steps during a conversation. Lead by the GUapp KG entities and categories, our conversational agent realizes dialogues deeply related to the recommendation domain. It also grants to manage two highly felt issues in the RecSys community, like the cold start problem and the absence of items to suggest.

5. Recommending Jobs through Dialogues

In this section, we present an example of the ontology-driven conversation with some details about how the knowledge base and the ontology are used in order to handle the negotiation. Given the conversational nature of the system, a preliminary step for preference elicitation is needed. In this phase, the agent asks the user some relevant information about her preferences. In particular, it asks about the geographical area and the field of interest. This first conversation phase follows a well-defined dialogue flow. Let us consider a scenario in which Claudio, a user who is graduated as Computer Engineer, is looking for jobs in the computer engineering field. Therefore, he visits the GUapp website and finds the section related to the conversational agent. After a standard welcome message, the agent asks for the geographical area Claudio is interested in. In this case, Claudio writes that he is interested to work in Rome. The second crucial question that the agent asks to Claudio is about the job position he would like to cover. He answers stating that he is interested in a job as Computer Engineer. At this point, the agent tries to map Claudio's responses to entities and class in the KG. In this particular case, the agent understands that Claudio is interested in jobs proposal in Rome (that is linked to the entity *gpr:roma*) regarding a position of computer engineer (which refers to the ontological class *gpo:computerEngineer*). As a result, by exploiting the Elasticsearch index, the agent starts searching for jobs that match the Claudio's requirements and creates a list of possible recommendations that perfectly match his preferences. The recommendations are ranked based on the relevance score described in section 3. Therefore, the agent provides only the first job proposals in the ranked list in order to not impact negatively the user experience. If Claudio asks for more results, the agent will explore the ranked list in order to provide other possible recommendations until he is satisfied or no more job proposal is available. If Claudio found an interesting job call, the interaction ends, otherwise the scenario is more interesting since the agent is not able to provide other solutions that perfectly match the user's needs. In that case, the agent needs more information to understand if the user is willing to travel or how flexible is with respect to the job place. We refer to this phase as *negotiation*. Figure 2 shows an example of the behavior of the agent in the negotiation and how it exploits the ontology for predicting other possible recommendations. At the end of the preference elicitation phase shown in the previous example, the agent creates a ranked list of possible job recommendations in the computer engineering field in Rome. Thus it provides to Claudio the first result that is job offer as a researcher offered by the University *La Sapienza* that is an instance of the *Computer Engineering* class in the ontology. Let us assume that Claudio does not find the job proposal interesting and he asks for more results. For the sake of simplicity, we assume that no more alternatives are available. In order to provide other solutions, the agent needs more information about Claudio.

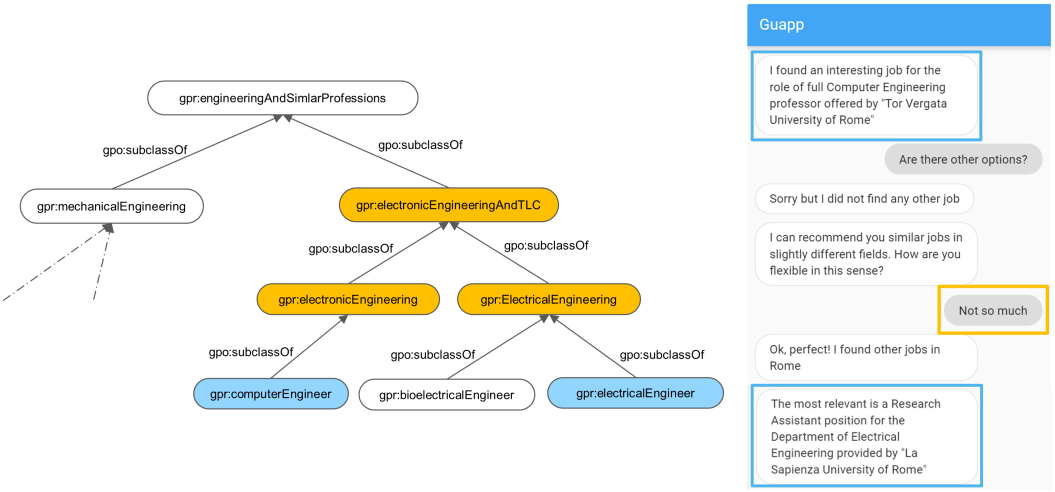

Figure 2: An example of Ontology driven negotiation. The messages in blue refer to the recommendations provided by the agent. The message in orange is the one that triggers the backward process in the ontology. Blue nodes in the ontology are the classes associated to the job proposals, while orange nodes are those explored in the negotiation phase.

In particular, since there could be possible job offers in slightly different fields, it asks about the user's flexibility with respect to the job place. Following the example, the agent asks Claudio *"I can recommend you similar jobs in slightly different fields. How are you flexible in this sense?"*. The agent maps the Claudio's answer to the number of edges that it could navigate backward in the ontology. For instance, if Claudio answers *"Not so much"*, it means that he is not interested in jobs that differ too much from the field he proposed previously. For this reason, the answer is mapped to a maximum of 2 backward hops in the ontology. In case Claudio replies with *"Quite flexible"*, the agent maps the answer to a number of 3 backward hops. This mapping has been empirically defined. Since our ontology is composed of 6 levels, 2 backward hops allow the system to provide job recommendations that belong to a similar domain as the previous ones. In this sense, the agents navigates backward the ontology starting from the class *gpo:computerEngineer* and reaching the parent node *gpo:electronicEngineeringAndTLC*. Starting from the inner node reached after the backward phase, a forward step is needed to reach the leaves of the sub-tree. At this point, following the example, the agent reaches the leaf *gpo:electricalEngineer* that, for simplicity, is the only node in the current sub-tree associated to possible job offers in Rome. Also in this case, it creates a ranked list following the same approach described previously and provides the user the most relevant alternatives.

The same method described in this section can be used in another possible scenario. Assuming that, during the negotiation phase, the user answers that she does not want the system to search for similar jobs or areas of work. In this case, the agent will ask if the user is willing to travel and, according to the answer, it will provide job proposals more or less distant from the original request. All the information about geographical entities is organized into the knowledge base. This allows the system to navigate the graph and find possible job offers that are in the same geographical area.

6. Conclusion and Future Work

In this paper we presented GUapp, a platform for searching jobs in the Italian public administration. We have outlined the architecture designed to make possible the job recommendation task in a conversational setting, besides describing the overall structure of the GUapp KG and its ontology. Thanks to that, GUapp consists of a recommender system that suggests relevant jobs to the users, and one of the most interesting aspects is the integration of a KG that helps driving the dialogue, making the interaction more natural and pushed at a finer grained level. Indeed, the preference elicitation becomes incremental, with the possibility of refining and improving the user requests. We are planning of introducing other several fine-grained features for making the ecosystem competitive with other state-of-the-art solutions, testing the User Experience with an A/B test and setting up a platform in order to share the dataset coming from GUapp usage with the industrial and academic community, with the respect of the privacy concerns.

References

[1] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender systems: an introduction, Cambridge University Press, 2010.

[2] V. Bellini, G. M. Biancofiore, T. D. Noia, E. D. Sciascio, F. Narducci, C. Pomo, Guapp: A conversational agent for job recommendation for the italian public administration, in: 2020 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2020, Bari, Italy, May 27-29, 2020, 2020, pp. 1–7. URL: https://doi.org/10.1109/EAIS48028.2020.9122756. doi:10.1109/EAIS48028.2020.9122756.

[3] S. T. Al-Otaibi, M. Ykhlef, A survey of job recommender systems, International Journal of the Physical Sciences 7 (2012) 5127–5142.

[4] J. Malinowski, T. Keim, O. Wendt, T. Weitzel, Matching people and jobs: A bilateral recommendation approach, in: Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06), volume 6, IEEE, 2006, pp. 137c–137c.

[5] R. Rafter, B. Smyth, Passive profiling from server logs in an online recruitment environment, in: Workshop on Intelligent Techniques for Web Personalization at the the 17th International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, August, 2001, 2001.

[6] R. Burke, Knowledge-based recommender systems, Encyclopedia of library and information systems 69 (2000) 175–186.

[7] W. Carrer-Neto, M. L. Hernández-Alcaraz, R. Valencia-García, F. García-Sánchez, Social knowledge-based recommender system. application to the movies domain, Expert Syst. Appl. 39 (2012) 10990–11000. URL: https://doi.org/10.1016/j.eswa.2012.03.025. doi:10.1016/j.eswa.2012.03.025.

[8] J. K. Tarus, Z. Niu, A. Yousif, A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining, Future Gener. Comput. Syst. 72 (2017) 37–48. URL: https://doi.org/10.1016/j.future.2017.02.049. doi:10.1016/j.future.2017.02.049.

[9] A. Bakarov, V. Yadrintsev, I. Sochenkov, Anomaly detection for short texts: Identifying whether your chatbot should switch from goal-oriented conversation to chit-chatting, in: International Conference on Digital Transformation and Global Society, Springer, 2018, pp. 289–298.

[10] L. Shang, Z. Lu, H. Li, Neural responding machine for short-text conversation, arXiv preprint arXiv:1503.02364 (2015).

[11] O. Vinyals, Q. Le, A neural conversational model, arXiv preprint arXiv:1506.05869 (2015).

[12] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, B. Dolan, A neural network approach to context-sensitive generation of conversational responses, arXiv preprint arXiv:1506.06714 (2015).

[13] J. Dodge, A. Gane, X. Zhang, A. Bordes, S. Chopra, A. Miller, A. Szlam, J. Weston, Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems, arXiv:1511.06931 [cs] (2015). URL: http://arxiv.org/abs/1511.06931.

[14] A. Bordes, Y.-L. Boureau, J. Weston, Learning end-to-end goal-oriented dialog, arXiv preprint arXiv:1605.07683 (2016).

[15] J. Dodge, A. Gane, X. Zhang, A. Bordes, S. Chopra, A. Miller, A. Szlam, J. Weston, Evaluating prerequisite qualities for learning end-to-end dialog systems, arXiv preprint arXiv:1511.06931 (2015).

[16] T. Zhao, M. Eskenazi, Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning, arXiv:1606.02560 [cs] (2016). URL: http://arxiv.org/abs/1606.02560, arXiv: 1606.02560.

[17] J. Williams, A. Raux, M. Henderson, The dialog state tracking challenge series: A review, Dialogue & Discourse 7 (2016) 4––33.

[18] B. Liu, I. Lane, An End-to-End Trainable Neural Network Model with Belief Tracking for Task-Oriented Dialog, Interspeech 2017 (2017) 2506–2510. URL: http://arxiv.org/abs/1708.05956. doi:10.21437/Interspeech.2017-1326, arXiv: 1708.05956.

[19] P. Wärnestål, User evaluation of a conversational recommender system, in: Proceedings of the 4th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, 2005, pp. 32–39.

[20] S. Haussmann, O. Seneviratne, Y. Chen, Y. Ne'eman, J. Codella, C. Chen, D. L. McGuinness, M. J. Zaki, Foodkg: A semantics-driven knowledge graph for food recommendation, in: The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II, 2019, pp. 146–162. URL: https://doi.org/10.1007/978-3-030-30796-7_10. doi:10.1007/978-3-030-30796-7_10.

[21] F. Narducci, P. Basile, M. de Gemmis, P. Lops, G. Semeraro, An investigation on the user interaction modes of conversational recommender systems for the music domain, User Modeling and User-Adapted Interaction (2019) 1–34. doi:https://doi.org/10.1007/s11257-019-09250-7.

[22] A. Iovine, F. Narducci, G. Semeraro, Conversational recommender systems and natural language: A study through the converse framework, Decision Support Systems (2020) 113250. doi:https://doi.org/10.1016/j.dss.2020.113250.