Adaptive Task-oriented Chatbots Using Feature-based Knowledge Bases

Carla Campàs[1], Quim Motger[1], Xavier Franch[1], and Jordi Marco[2]

[1] Department of Service and Information System Engineering, Universitat Politècnica de Catalunya
{jmotger,franch}@essi.upc.edu, carla.campas@upc.edu
[2] Department of Computer Science, Universitat Politècnica de Catalunya
jmarco@cs.upc.edu

Abstract. Task-oriented chatbots relying on a knowledge base for domain-specific content exploitation have been largely addressed in research and industry applications. Despite this, multiple challenges remain to be fully conquered, including adaptive knowledge mechanisms, personalization for user-specific demands, and composite intent resolution. To address these challenges, in this paper, we present a work-in-progress summary of a task-oriented, knowledge-based chatbot in the field of mobile software ecosystems. The chatbot is designed to assist users in the combined use of multiple features from different applications. The proposed knowledge base and the machine learning pipeline supporting the chatbot technical core are designed to: (i) effectively use user context, (ii) process runtime feedback, (iii) use user historical data, and (iv) automatically infer slot values and dependent actions. With this report, we expect to lay the groundwork for future development stages and user validation studies.

Keywords: Natural Language Understanding · Chatbots · Mobile Feature Integration · Adaptive Knowledge Bases

1 Introduction

Knowledge-based dialogue systems (i.e., chatbots) define content-based interactions with users through the integration of multiple, heterogeneous data repositories to build a centralized information system (i.e., a knowledge base) for a given domain [10]. Overall, this knowledge bases support: (1) entity linking or extraction (i.e., the relation between entity mentions in user utterances and their corresponding entities in the knowledge base), and (2) intent classification (i.e., the relation between user requests and the resources or entities in the knowledge base that the chatbot can use to generate an appropriate response). While both of these tasks have been intensively addressed in the Natural Language Understanding (NLU) field, several challenges remain yet to be fully conquered. In the field of task-oriented chatbots (i.e., designed to execute a particular action from a known sub-set of pre-configured tasks [6]), one of the main challenges is the design of adaptive, personalized knowledge bases modelling the unique needs of a user, in order to provide a customized (and therefore, enhanced) experience when interacting with the chatbot [1]. Effective use of the user context, processing runtime feedback, and accessing user historical data can enact updates in the knowledge base by modelling the extracted knowledge into its entities and relations. Another significant challenge is the recognition of composite user intents, where a single intent is decomposed into multiple actions and requires extended knowledge generation to complete and respond to a specific user utterance [7]. In this case, the information embedded in the knowledge base can be used to infer slot values (i.e., entity values for a parameterized intent) and dependent actions (i.e., subsequent requests to third-party software systems) [2].

In this paper, we address these challenges by presenting the design and initial implementation stages of a knowledge-based, task-oriented dialogue system in mobile software ecosystems. Through an adapted design science research method, we address the following research question: **RQ. How task-oriented, knowledge-based chatbots can effectively and efficiently integrate adaptive, personalization, and feedback-aware mechanisms?** The resulting chatbot is designed to consume a knowledge base modelling a catalog of mobile apps, the set of functionalities (i.e., features) these apps expose, and the parameters (i.e., data items) used and generated by these features. Its main goal is to provide an adaptive, customizable experience to end users in the execution and integration of these features. Overall, this research aims at laying the groundwork for future research in the field of adaptive, context-aware, and personalized generation of knowledge-based, task-oriented dialogue systems.

2 Conceptual model and knowledge base

Figure 1 presents a high-level instance of the conceptual model for mobile software ecosystems used to design the chatbot knowledge base. This model is a feature-oriented extension of the mobile software ecosystem model as defined in [9]. We briefly describe each entity, how they relate to each other, and how these concepts align with the challenges and research proposal (see Section 3).

- **User.** An individual end-user of a mobile device (e.g., *Alice*).
- **App.** A mobile-based software application (e.g., *Strava*). The set of apps used by a single user is what constitutes their own application portfolio (e.g., Alice's portfolio is composed of Strava and Google Calendar).
- **Feature.** A functional requirement (from the user perspective) exposed by an app (e.g., *PlanRoute*). A single feature can be exposed by multiple apps (e.g., *ScheduleEvent* is exposed by Google Calendar and Simple Calendar).
- **Parameter.** A type-based attribute defining a property-value pair for a given feature (e.g., *start-location* in *PlanRoute*).
- **Feature Integration.** A combined, semi-automatic use of two independent features (e.g., *PlanRoute-ScheduleMeeting*), defined by the integration of a *source* feature (e.g., *PlanRoute*), which acts as a trigger for the feature integration, and a *target* feature (e.g., *ScheduleMeeting*), which acts as the

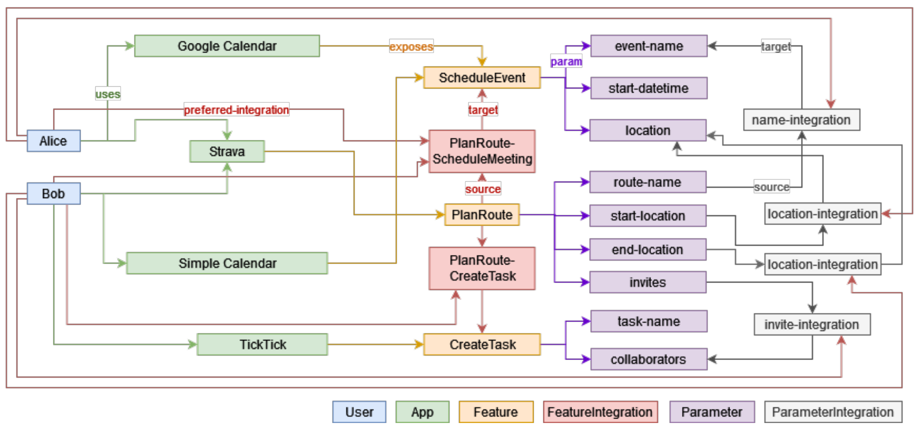

Fig. 1. Example instance of the knowledge base

outcome, automated enacted feature based on the trigger feature. A feature integration facilitates a user's combined use of multiple features to achieve a major, complex goal. For instance, when Alice plans a route with the Strava app, the system supports scheduling an event with the Google Calendar app. This feature integration proposal is user-dependent, which means that each user defines their own *preferred-integration* relations between a source feature and a target feature between two specific apps in their own portfolio. For instance, Alice might schedule an event in Google Calendar from a newly planned route in Strava, while Bob might not only create an event in a different app from his own portfolio (e.g., Simple Calendar), but also he might create a new task with another app (e.g., TickTick).

– **Parameter Integration.** Implementation of a parameter-based integration between a source feature parameter (e.g., *start-location* from *PlanRoute*), and a target feature parameter (e.g., *location* from *ScheduleEvent*). A parameter integration facilitates to users the exchange of required data between multiple features to enact a feature integration.

The main goal of the chatbot is to act as a facilitator on the integration of features defined by the conceptual model. Through dialogue-based communication, the chatbot consumes the knowledge base to (1) trigger the corresponding feature, (2) adapt and customize the conversation with users according to their preferences, (3) collect additional, missing data from the user to run the feature, and (4) to enact any required updates to the instance knowledge-based.

3 Chatbot story design

To illustrate the chatbot's expected behaviour, we design a set of stories (i.e., a conversation scenario between the user and the chatbot) composing the dialogue management module. Our contribution focuses on the following design dimensions, supported by the knowledge base, as exemplified in Figure 1:

1. **Use of user context (C1).** The proposed knowledge base models various context dimensions from mobile software ecosystems, including user information (e.g., app portfolio) and preferences (e.g., preferred feature integrations) as defined by Daniel (2018).
2. **Process runtime feedback (C2).** The chatbot is responsible for analysing positive and negative feedback from the user to start/stop a specific feature or parameter integration and enact any required updates in the knowledge base to reflect the user context and preferences (e.g., preferred integrations).
3. **Use historical data (C3).** Historic data about enacted integrations and the conversation tracker history between the chatbot and the user can be used to query and adapt user-specific customization details on the knowledge base. Additionally, the conversation history can be used to re-train entity extraction and intent recognition tasks.
4. **Infer parameter (slot) values and dependent actions (C4).** Either through preferred parameter and feature integrations modelled by the knowledge base or through traditional slot extraction (i.e., entity extraction and intent recognition) from the conversation with the user.

Figure 2 shows a generic version of the storylines as depicted below.

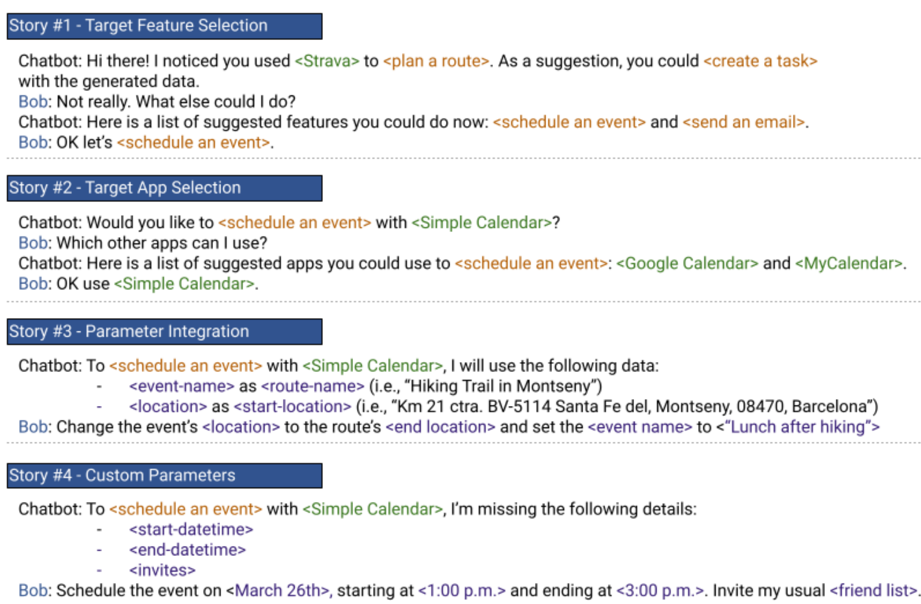

Fig. 2. Chatbot main storylines for task-oriented dialogue management

Target Feature Selection. From a source feature in the user's device (i.e., trigger for a feature integration) the chatbot queries the knowledge base for potential feature integrations based on the user's device feature. It then selects the best integration for the user by querying their preferred integration history (C3). Finally, it asks the user if they want to proceed with the integration. The user has mainly 3 options (C2): (1) to accept the feature integration proposal; (2) to deny the proposal and finish the conversation; or (3) to request new integration suggestions (as in Story #1, Figure 2).

Target App Selection. For a given target intent, the chatbot requires a target app to enact the feature. Similarly to target feature selection, the apps suggested by the chatbot are queried from the knowledge base through the *preferred-integration* (C3) relations (updated through historic data and user feedback (C1)). Once again, the user might respond (C2): (1) to accept the application selection; (2) to refuse the selection and end the process; or (3) to request application suggestions for integration (as in Story #2, Figure 2).

Parameter Integration. For a given feature integration, the chatbot must determine what information from the source intent can be parsed to the target intent (C4). The knowledge base contains information on which of these parameters can be reused (i.e., parameter integration). The chatbot presents the suggested parameter integrations for user validation. Overall, the user might want: (1) to accept all suggested parameter integrations; or (2) to recursively change and refactor parameter integrations with different source parameters or with custom values (as in Story #3, Figure 2). Consequently, if desired, the user has the option to re-arrange the information in different ways.

Custom Parameters. Given that the target intent requires more data or has any optional fields the user might want to fill in, the chatbot attempts to recompile the remaining information from the user (i.e., parameters for which there is no explicit integration). For each remaining parameter, the user might want: (1) to input custom values for that specific feature integration enactment (as in Story #4, Figure 2); or (2) to reference source intent parameters, suggesting a new potential parameter integration. Similarly to the parameter integration story, the user can recursively change the input data until final validation.

4 Component Design and NLU Pipeline

Figure 3 shows the proposed, work-in-progress machine-learning pipeline for the chatbot aiming at supporting the stories depicted in Section 3. The chatbot[3] is being developed using the RASA framework[4].

User intent pre-processing. As a first step, the pipeline processes the user intent through multiple state-of-the-art, standard NL-based components before

[3] Available at: https://github.com/gessi-chatbots/knowledge_based_chatbot

[4] RASA: Open-source conversational AI framework: https://rasa.com/

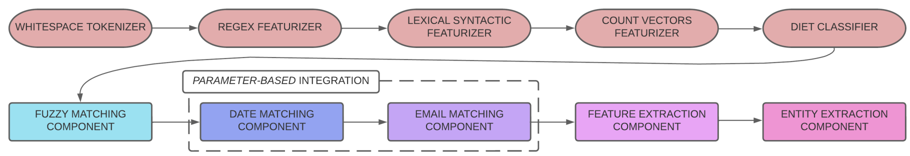

Fig. 3. Chatbot NLU Pipeline

feeding the user intent to the custom pipeline components. These include: (1) a tokenizer; (2) multiple feature extraction components (e.g., regex-based, lexical, syntactic, sentence embeddings); and (3) a Dual Intent Entity Transformer (DIET) for intent classification and entity extraction [3].

Custom NLU pipeline. To address the specific challenges and dialogue management requirements from our design, we have extended the NLU pipeline with multiple customized, adapted components. Mainly:

- **Fuzzy matching.** Embedded the fuzzy matching technique for effective resolution of spelling mistakes and deviations from the user input [8], to avoid potential mismatches between the user intents (i.e., feature and parameter recognition) and the knowledge base modelled data.
- **Parameter-based extraction.** Prior to model-based entity extraction, we extended the pipeline with rule-based extraction techniques for standard data types (e.g., dates, e-mails) from our parameter data model (e.g., regular expressions, custom classifiers).
- **Feature extraction.** To find source and target features, we extended the pipeline with a pre-trained language model fine-tuned for feature extraction using crowdsourced data in the mobile app domain (C4). We conducted initial fine-tuning experiments with BERT [4] and T5 [12], which have shown success in abstractive summarization. The pre-trained model extracts intended features from user input. If the intent requires application features, the system can provide some from the user text.
- **Entity extraction.** We aim to enhance the pipeline by incorporating a named-entity recognition model to extract particular data types from the knowledge base parameters data model (e.g., names of individuals, organizations, phone numbers, addresses), similar to the feature extraction process (C4). For composite intents (i.e., referring to multiple parameters), we exploit the attention mechanism provided by the transformer sentence embeddings to disambiguate between multiple parameters with a shared data type (e.g., start/end date in Figure 2).

5 Planned evaluation

The evaluation plan is mainly structured into two sub-sets of experiments and evaluation tasks. The first one is oriented to the technical verification of the

NLU pipeline and the specific components used for dialogue management, and intent and entity extraction tasks. We plan to build a validation set of annotated documents (e.g., documents, texts, user intents) which will be used for evaluating these components separately. Through a cross-validation analysis technique, this evaluation will be focused on accuracy, precision, recall, and f-measure metrics, as well as an analysis of the models' performance. Additionally, we plan to focus on the analysis of the conversation flow and the tracking of user feedback.To this end, we plan to conduct user validation studies, which will allow us to monitor and analyse user interactions with the chatbot. On one hand, we plan to focus on conversation flow analysis, to evaluate the compliance of the stories with the user's expectations. This will allow us to see deviations from default story paths or repeated errors, as well as the confidence with which our models predict intents, capture entities and identify features. On the other hand, processing user feedback will provide a general overview in terms of user satisfaction, user adherence, and learnability of the system (among others) to assess the quality degree of the stories as designed.

6 Related Work

Composite task-oriented chatbots and adaptive knowledge bases are undergoing intense study as a consequence of the latest innovations in the field of natural language. Concerning the former, Bouguelia et al. proposed a knowledge system for composite intent resolution in task-oriented chatbots [2]. While effective for the task at hand, they did not consider using user historical data processes or adaptive content techniques for the knowledge system. In this sense, Qin et al. proposed a knowledge base query by combining user intent with a given dialogue history [11], but with no specific adaptive content strategy. Xue et al. effectively applied user feedback to retrain models for a given NLU pipeline [15]. Similarly, Sapna et al. extended this model retraining task with user data from third-party software systems (i.e., external to the chatbot) [14]. Concerning adaptive knowledge bases, Raghu et al. proposed a language-resolution-based technique for knowledge base adaptations to improve user adherence [13]. Huang et al. designed a crowd-powered approach based on user feedback and focused on user stories and chatbot responses, similarly towards the same objective [5]. Overall, these studies provide a clear perspective on related work addressing a specific challenge from the design dimensions addressed in this paper.

7 Conclusions and Future Work

This paper introduces a task-oriented, knowledge-based chatbot to support the semi-automatic integration of multiple features in the context of mobile software ecosystems. Its design is oriented to illustrate the applicability of adaptive, personalized experience mechanisms and complex intent and entity resolution techniques. The following steps will require (i) the latest development stages of the machine learning pipeline components, (ii) design of the knowledge base query operations and integration of the chatbot with the knowledge base, and (iii) experimentation for verification of the custom machine learning components and validation of the conversational process with users.

Acknowledgments

With the support from the Secretariat for Universities and Research of the Ministry of Business and Knowledge of the Government of Catalonia and the European Social Fund. This paper has been funded by the Spanish Ministerio de Ciencia e Innovación under project / funding scheme PID2020-117191RB-I00 / AEI/10.13039/501100011033.

References

1. Bavaresco, R., et al.: Conversational agents in business: A systematic literature review and future research directions. Computer Science Review **36**, 100239 (2020)
2. Bouguelia, S., et al.: Context Knowledge-Aware Recognition of Composite Intents in Task-Oriented Human-Bot Conversations. In: Advanced Information Systems Engineering. pp. 237–252. Springer International Publishing (2022)
3. Bunk, T., et al.: DIET: Lightweight Language Understanding for Dialogue Systems (2020). https://doi.org/10.48550/ARXIV.2004.09936
4. Devlin, J., et al.: Bert: Pre-training of deep bidirectional transformers for language understanding (2018). https://doi.org/10.48550/ARXIV.1810.04805
5. Huang, T., Chang, J., Bigham, J.: Evorus: A crowd-powered conversational assistant built to automate itself over time. Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (2018)
6. Hussain, S., Ameri Sianaki, O., Ababneh, N.: A survey on conversational Agents/Chatbots classification and design techniques. In: Web, Artificial Intelligence and Network Applications (2019)
7. Jain, M., et al.: Evaluating and informing the design of chatbots. In: Proceedings of the 2018 Designing Interactive Systems Conference. p. 895–906 (2018)
8. Kalyanathaya, K., Akila, D., Suseendran, G.: A Fuzzy Approach to Approximate String Matching for Text Retrieval in NLP. Journal of Computational Information Systems **15**, 26–32 (05 2019)
9. Motger, Q., Franch, X., Marco, J.: Integrating Adaptive Mechanisms into Mobile Applications Exploiting User Feedback. In: Research Challenges in Information Science. pp. 347–355 (2021)
10. Motger, Q., Franch, X., Marco, J.: Software-based dialogue systems: Survey, taxonomy, and challenges. ACM Comput. Surv. **55**(5) (2022)
11. Qin, L., et al.: Entity-consistent end-to-end task-oriented dialogue system with KB retriever. In: EMNLP-IJCNLP 2019. pp. 133 – 142 (2019)
12. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer (2019). https://doi.org/10.48550/ARXIV.1910.10683
13. Raghu, D., Gupta, N., Mausam: Disentangling language and knowledge in task-oriented dialogs. In: NAACL HLT 2019. vol. 1, pp. 1239 – 1255 (2019)
14. Sapna, et al.: Recommendence and fashionsence: Online fashion advisor for offline experience. CoDS-COMAD '19 (2019)
15. Xue, Z., et al.: Isa: Intuit smart agent, a neural-based agent-assist chatbot. In: 2018 IEEE International Conference on Data Mining Workshops (ICDMW) (2018)