Neurocomputing 271 (2018) 59–69

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Building multi-domain conversational systems from single domain resources

David Griol*, José Manuel Molina

Applied Artificial Intelligence Group, Computer Science Department, Carlos III University of Madrid, Avda. de la Universidad, 30. 28911 - Leganés (Spain)

ARTICLE INFO

Article history:
Received 8 May 2016
Revised 14 September 2016
Accepted 29 January 2017
Available online 6 July 2017

Keywords:
Dialog systems
Multi-domain
Spoken interaction
Human-machine interaction
Neural networks
Statistical methodologies

ABSTRACT

Current advances in the development of mobile and smart devices have generated a growing demand for natural human-machine interaction and favored the intelligent assistant metaphor, in which a single interface gives access to a wide range of functionalities and services. Conversational systems constitute an important enabling technology in this paradigm. However, they are usually defined to interact in semantic-restricted domains in which users are offered a limited number of options and functionalities. The design of multi-domain systems implies that a single conversational system is able to assist the user in a variety of tasks. In this paper we propose an architecture for the development of multi-domain conversational systems that allows: (1) integrating available multi and single domain speech recognition and understanding modules, (2) combining available system in the different domains implied so that it is not necessary to generate new expensive resources for the multi-domain system, (3) achieving better domain recognition rates to select the appropriate interaction management strategies. We have evaluated our proposal combining three systems in different domains to show that the proposed architecture can satisfactory deal with multi-domain dialogs.

1. Introduction

A spoken dialog system (SDS) is a software that accepts natural language as an input and produces natural language as an output engaging in a conversation with the user [1–3]. To successfully manage the interaction with users, spoken dialog systems usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialog management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules.

Recent advances in conversational interfaces has been propelled by the convergence of three enabling technologies. First, the Web emerged as a universal communications channel. Web-based dialog systems are scalable enterprise systems that leverage the Internet to simultaneously deliver dialog services to large populations of users. Second, the development of mobile technologies and intelligent devices, such as smartphones and tablets, have made it possible to deploy a large number of sensors and to integrate them into dialog systems that provide multimodal interaction capabilities (i.e., use of different modalities for the input and/or output of the system) and allow their access in almost every place and at any time. Third, Computational Linguistics has significantly increased speech recognition, natural language understanding and speech synthesis capabilities [1,3,4].

These advances have extended the initial application domains of dialog systems to complex information retrieval and question answering applications [5], surveys applications [6], e-commerce systems [7], recommendations systems [8], e-learning and tutoring systems [9], in-car systems [10], spoken dialog within vehicles [11], remote control of devices and robots in smart environments [12], Ambient Assisted Living systems [13], or virtual companions [14].

However, spoken dialog systems are usually employed within single domains, which are also defined according a static set of strong restrictions. In mobile environments, the dynamic support for a wide range of topics and multiple tasks within one and the same dialog is still a major challenge as people require increasingly more functionalities in the system [15,16].

In this paper we contribute a novel approach to generate multi-domain conversational systems that are able to hold a conversation in which the user switches from a domain to the other. Section 2 presents a state of the art of the most relevant

* Corresponding author.
E-mail addresses: david.griol@uc3m.es, dgriol@inf.uc3m.es (D. Griol), josemanuel.molina@uc3m.es (J.M. Molina).

http://dx.doi.org/10.1016/j.neucom.2017.01.112

approaches that exist to address this challenge and how they relate with our proposal. Section 3 describes our proposal in detail. Section 4 presents the experimental set-up to show the applicability of our proposal. Section 5 discusses the results of the evaluation for the developed multi-domain dialog system, and finally Section 6 presents conclusions and outlines possibilities for future work.

2. State of the art

The most widespread approach to develop multi-domain dialog systems is based on the use of distributed architectures [17–19]. These architectures are based on using a single interface with specific dialog managers to deal with each domain or set of related tasks. Selecting the most appropriate domain is then one of the key technologies to develop a multi-domain dialog system [16].

To solve the domain identification problem in a multi-domain dialog system, the initial approach was asking users to explicitly specify the domain [20]. Although this approach prevents the problem of ambiguity caused from polysemous words, the dialog strategy is not natural and users have to know each domain in advance [21]. The distributed architecture described in [19] proposes domain identification computing the most similarity scores of the recognized user utterance and the grammar/language model of domains. Although it is an implicit domain identification approach, the identification of ambiguous domains is still not solved because the similarity scores could be almost the same for polysemous words.

As an alternative to these initial approaches, pre-selection or post-selection methodologies have been proposed for domain selection. Pre-selection methodologies select the most appropriate dialog manager by considering the features extracted from the user utterance. In this approach, a specific module is included to parse the user turns and redirect them to the appropriate single-domain dialog manager [17,19,22]. This approach is efficient in execution time, but requires incorporating domain-specific knowledge to improve the domain selection process [16,23].

The domain selection process proposed in [15] is based on a Logistic Regression (LR) model to classify the current user turn into a specific domain. The features used for the vectorization of the sentence are: bag of words, bag of bigrams, and co-occurrence of two words in the same sentence. A one-vs-all classifier was created for each subtask to assign a score to the input sentence. The task with highest confidence is usually selected. If the confidence of the second-ranked task is close, a Task Manager outputs a disambiguation turn asking the user to add some domain-related expression.

A series of domain-independent analyzers including linguistic analysis, generic spoken language understanding (SLU) analysis, and keyword analysis is proposed in [16]. Based on the analyzed results, domain selection is performed by two-step approaches: domain ordering and domain filtering. In the domain-ordering step, the domain candidates are listed in descending order of scores computed by a pre-selection model. Then, content-based domain filtering is performed for each domain in order to determine the selected domain.

A two-level SLU approach is proposed in [24], where, at first level, the result of ASR is tagged and classified by general SLU for domain spotting by employing a maximum entropy-based classifier using lexical word, dialog acts and previous domain as classification features. The use of two ontologies is proposed in [25]: one associated with a broad-coverage SLU module, and a second associated with a task-domain. The use of Recurrent Neural Networks has been very recently proposed to complete this task in [26,27].

Post-selection methodologies are based on the results provided by the different dialog managers integrated in the multi-domain system [28]. The main advantage of this approach is that rich domain-specific features are considered to improve performance. However, executing all the dialog managers on the unrelated domains can be a waste of time especially when the number of domains increases [16].

Gasic et al. have recently proposed a distributed multi-domain dialog architecture in which dialog policies are organized in a class hierarchy aligned to an underlying knowledge graph [29]. Gaussian process-based reinforcement learning is proposed to construct generic dialog policies. A policy committee model, based on a Bayesian committee machine (BCM) is also proposed in [30] to further improve the performance when the training data is limited.

Some techniques for multi-domain semantic speech recognition and understanding and language generation have been presented during recent years [31–35]. In the case of dialog management, the exponential increase of dialog states makes it really difficult to create a dialog manager that can serve several domains using the same approaches that are used for single-domain systems [15].

As described in [21], there are two major difficulties for multi-domain dialog management. The first one is to interpret users' interested domain correctly given ambiguous user utterances across different domains. The second one is the high cost of merging the dialog management of different single-domain systems into one multi-domain system.

Frame-based representations are usually employed to model the semantic representation of the user' contents in recent proposals to develop multi-domain dialog systems [23,28,36]. The form interpretation algorithm (FIA), the basis for the VoiceXML standard[1], is an example of a model of frame-based dialog management. The use of this standard is proposed in [37] to implement multi-domain dialog systems in which a content manager automatically extracts the contents for each domain from the Internet, and a content spotter selects the most appropriate source by means of the cosine similarity function.

The "Information State" theory [38] represents a dialog by means of the information required to differentiate it from other dialogs. This information, which is also referred as the "discourse context" or the "mental state" represents the effects of the sum of previous actions during the dialog and the motivation for the future system actions. This approach was used to develop the GodiS architecture, which is proposed in [39] to develop multi-domain dialog systems. This approach is also adopted in [40] to develop a multi-domain SDS that uses a discriminative classification model for more accurate state updates.

Agent-based dialog management approaches combine the benefits of finite-state and frame-based dialog management approaches [41]. The dialog managers developed by means of this approach allow to execute and monitor operations in dynamically changing application domains. The RIME framework (Robot Intelligence based on Multiple Experts) employs this approach to integrate different dialog agents, which are specialized in achieving specific tasks by means or performing physical actions or engaging the user in a dialog [42].

Different example-based dialog management approaches have been recently proposed to develop multi-domain dialog systems [23,43–45]. The dialog managers developed by means of these approaches employs a database of pairs of a dialog example and the corresponding system action. As described in different contributions [23,46,47], the dialog managers developed by means of this approach can be easily and flexibly modified by updating dialog examples in a database. This is specially important to construct multi-domain dialog managers when the specific domains or tasks can be frequently expanded or when the limited knowledge about the task makes very difficult to define rule-based managers.

[1] https://www.w3.org/TR/voicexml20/.