Chatbots - keeping track of context

Carl Balmer
University of Bern
Matr. Nr.: 13-120-431
Email: carl.balmer@students.unibe.ch

Mathias Fuchs
University of Bern
Matr. Nr.: 09-923-764
Email: fuchsmat@students.unibe.ch

***Abstract*—Nowadays chatbots become more and more sophisticated conversationalists, due to recent advances in the field. Chatbots are especially popular in handling customer service tasks. However it is crucial for a bot to be able to keep the context of a conversation. In this paper we give an overview over the different ways of contexts, the current state of the art in context tracking and we test a nerual network approach in an experiment, using the ubuntu dataset [1].**

I. Introduction

As the popularity of chatbots increases it becomes more important to increase their quality. This is why it is crucial for a chatbot to be able to keep track of the context. For example should the bot be able to know the nationality of the person using it or whether a person means his mother when saying "she" or his wife.

There are different types of context: The world knowledge(time, location, weather) , the user knowledge(relationships, preferences) and the dialogue context(Knowledge learned during the conversation), which is also called dialogue state. (need citation here?). In the following sections we give a brief overview over all those types. However our main focus will be on the dialogue context and the most common ways used to track it.

There are different approaches used for dialogue context:

- Rule-based approaches
- Probabilistic approaches
- Data driven approaches

Depending on which domain we are in, different approaches are more suited. There are two domains:

- Open domain
- Closed domain

II. Types of context

A. World knowledge and User knowledge

In order to be fully operable, a bot should always know the location, the time, the weather etc. of where the user is. Because what good is the chatbot if he only proposes restaurants that are in Australia, when the user is living in Switzerland? Or if the user asks about the local weather, he gets the weather in India. We called this knowledge "World knowledge". The bot should also know about the personal preferences, relationships, schedules etc. of the user. We called this knowledge "User knowledge". The cultural differences can also be huge. For example it has been found that Indians take a lot of time to articulate their intent and therefore prefer typing over talking [3].

Hence it is of great importance for a conversational agent to take into account the surroundings of the user, in order to be able to give appropriate answers.

B. Dialogue context

Dialogue context, also called dialogue state, represents the challenge of keeping track of the intent of the user and the knowledge learned during the conversation. For example:

> **U1:** Person: I would like to see the best italian restaurant
> **U2:** Conversational Agent: Hey, it is "luigis pizza". Would you like to make a reservation?
> **U3:** Person: No, please show me Chinese restaurants
> **U4:** Conversational Agent: We have "Restaurant A", "Restaurant B" and "Restaurant C" which are close. Which one do you like?
> **U5:** Person: Ok I've changed my mind. Make a reservation at the Italian place

In the above example the conversational agent needs to know what the user means in U5 by "the Italian place". It has to realize that the user is referring to the Italian restaurant mentioned in U2. Depending on how far back the context goes, it can be hard to track. Often the chatbot needs to ask again what the user means, because it did not understand.

III. The context tracking Problem

We can define the general case of the problem as follows:
$[u_1, a_2, u_3, a_4, u_5, ..., u_t] \rightarrow a_{t+1}$
Where

- u_i are the user actions which consist of stating an intent, give information, provide feedback to agent answers or change the intent or the given information.
- a_j are the agent actions which consist of requesting more information, give appropriate responses and request feedback.

In this case the agent decides for the appropriate response based on **all** previous utterances.

If we look at the Problem of Dialogue State Tracking in a general case, we can formulate it like this: The agent takes as input all previous utterances by both parties and then decides based on this, what the most appropriate response is. At any given point in the conversation the user and the agent both

[1] http://dataset.cs.mcgill.ca/ubuntu-corpus-1.0/

can take multiple actions. The user may state his Intent and/or give additional information (e.g. I want to ride the bus from → to). The agent can then reply by requesting additional information or presenting the user with a response and may ask for feedback

The hard thing is to track all of the information and notice changes in the users intent, the information and to incorporate feedback.

A. Closed vs. open domain

Before we talk about different approaches and methods to tackle this problem, we quickly have to mention the importance of domain ”size” because this greatly influences the feasibility and effectiveness of different methods.

There are two domains [1] in which a chatbot can operate in.

- **Closed domain**: The bot has one specific, well defined task it has to perform e.g. music player, restaurant finder. The advantage here is, that all the possible actions are finite and known in advance [2] (e.g. there is a finite number of Italian restaurants in a certain area). This means that it would for example be possible to model the agent as a set of states with transitions. Rule-based and probabilistic approaches are better suited/ most widely used in a closed domain.
- **Open domain**: The bot has no specific task. In the most extreme case it is completely open, which means that it has to converse with anyone about anything. This makes it impossible to model a set of possible answers beforehand. In this domain data-driven approaches are most widely used.

Fig. 1. Arrow representation of closed vs.open domain

As Figure 1 shows, we have made the observation that for a closed domain the rule based methods work well and the more open the domain is - the harder it gets - the more deep learning is used.

These are no hard borders, as most chatbots fill fall somewhere along this axis (e.g. Helpdesk). But in general we can say that the task gets more difficult, the more open a domain is. For the closed domain we can easily use rule-based approaches and for open domain deep learning makes more sense.

IV. State of the art approaches

In this section we are taking a look at the most commonly used methods to keep track of context in a conversation.

A. Rule-based approaches

This is the most simple form of keeping track of the context. Most commercial systems use a hand-crafted approach [**?**]. A fixed set of heuristics decides what to do in which situation. The given input is always matched to **one** specific output state.

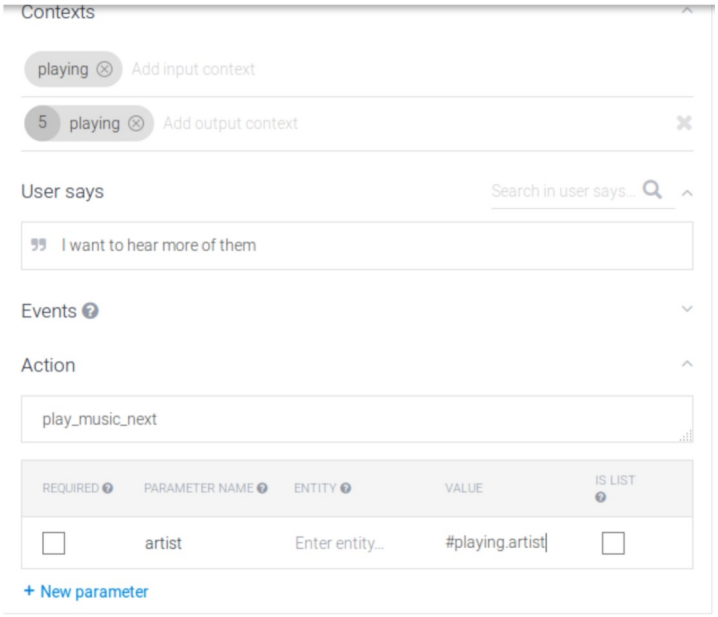

Fig. 2. Example of a rule-based approach from www.dialogflow.com

In Figure 2 we can see an example of a rule-based approach. How it works is, that if the user says something similar to ”I want to hear more of them”, then it recognizes the input context as ”playing music” and it will start playing more music of the artist that is currently playing.

Of course this whole process is impossible in an open domain, because the possible set of conversation status is infinitely and we can not create an infinite amount of hand crafted rules to match it and the rules would fail to work for unexpected utterances [**?**].

B. Probabilistic approaches

One of the most common and most successful approaches in this field is the hidden information state (HIS) approach to dialogue management [**?**] [**?**]. It is a variation of the classic partially observable markov decision process(POMDP) [**?**] model [**?**].

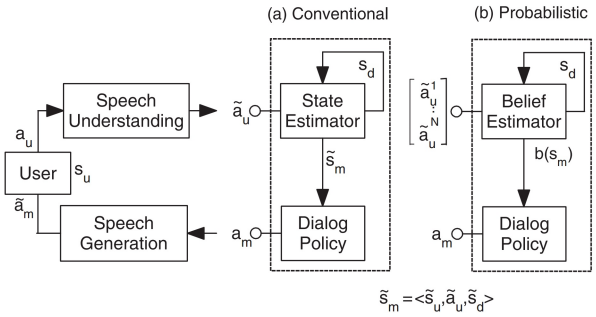

Fig. 3. Partially observable markov decision process (POMDP) based spoken dialogue system.

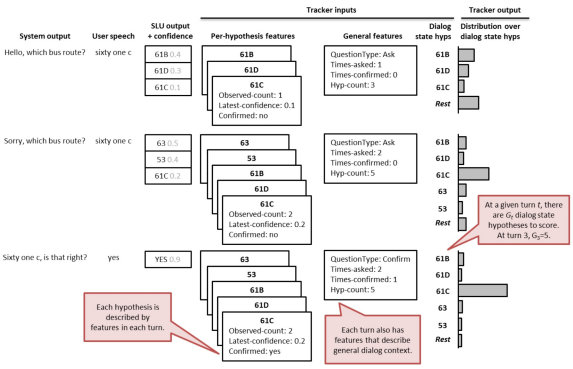

Fig. 5. Example of probabilistic dialogue state tracking process [**?**].

DESCRIBE POMDP model, how muhc? This approach assumes that dialogue evolves as a Markov process, i.e., starting in some initial state s_0, each subsequent state is modeled by a transition probability: $p(s_t|s_{t-1}, a_{t-1})$. The state s_t is not directly observable reflecting the uncertainty in the interpratetion of user utterances; instead, at each turn, the system regards the output of the SLU as a noisy observation o_t of the user input with probability $p(o_t|s_t)$. The transition and observation probability functions are represented by a suitable stochastic model, called here the dialog model M. The decision as to which action to take at each turn is determined by a second stochastic model encoding the policy P. As the dialog progresses, a reward is assigned at each step designed to mirror the desired charactersitics of the dialog system. The dialog model M and the policy model P can then be optimized by maximizing the expected accumulated sum of these rewards either online through interaction with users or offline from a corpus of dialogs collected within a similar domain.

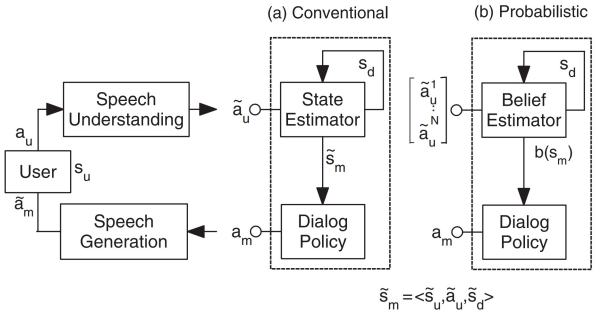

Fig. 4. Conventional model vs. probabilistic POMDP model [**?**].

Describe what pomdp vs traditional does

Describe DST graph

[**?**] [**?**]

C. Data driven approaches

In a open domain it quickly becomes impossible to define all possible states and state transition. It is therefore not possible to handcraft a system which solves the context state tracking problem.

But with the rise of big data and the large amount of conversations happening on the web, we can use data driven approaches to circumvent this problem. Instead of handcrafting a the mapping from context to response, we can learn this mapping from the data. reference to formal definition

In general there are two conrasting data driven approaches; *Discriminative and Generative*. For both approaches there are statistical and deep learning methods available.

1) Discriminative methods: The discriminative approach is to have a (very) large database of predefined responses and then pick the most appropriate one based on the context. This basicly reduces the task to a retreval problem akin to search engines (retreaving results based on a querry).

Most discriminative methods use the data to learn a ranking function. This function is then used to score all possoble responses. The best scoring response is given as an answer to the user.

An example for a statistical method ist *Okapi BM25*. A search engine ranking function based on ”bag-of-words” retreaval.

Is is also possible to train an artificial neural network to score replies. This method is further explained in section insert section, as we further investigate this method in our ablation study.

Discriminate

2) Generative methods:

V. Ablation Study

A. Goal

B. Dataset

C. Model architecture

D. Methology

E. Results

VI. Conclusion

The conclusion goes here. [3]

References

[1] R. Yan, Y. Song, X. Zhou, and H. Wu, "Shall i be your chat companion?: Towards an online human-computer conversation system," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 649–658.

[2] F. Radlinski and N. Craswell, "A theoretical framework for conversational search," in *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. ACM, 2017, pp. 117–126.

[3] S. Chopra and S. Chivukula, "My phone assistant should know i am an indian: influencing factors for adoption of assistive agents," in *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 2017, p. 94.