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 (in the following: conversational agent/ agent) become more and more sophisticated conversationalists, due to recent advances in the field. Conversational agents are especially popular in handling customer service tasks. However it is crucial for a conversational agent to be able to keep the context of a conversation. In this paper we first give an overview over the different types of contexts and the current state of the art in context tracking. Finally we perform an ablation study to investigate if neural networks do learn to use the context information given to them. Our experiments show that this is indeed the case.**

I. Introduction

As the popularity of conversational agents increases it becomes more important to increase their quality. This is why it is crucial for an agent to be able to keep track of the context. We split up the various forms of context [1], [2] in three different types:

- The world knowledge (time, location, weather). Example: The agent should know the local time of the user, to be able to provide opening times of shops.
- The user knowledge (relationships, preferences). Example: The agent should know whether a person is referring to his mother or his wife, when he is using the word "she".
- The dialogue context or dialogue state tracking (knowledge learned during the conversation). Example: *Agent:* We have this fancy Italian place. *User:* Ok, make a reservation for it.

The main focus of this paper will be on keeping track of the dialogue context.

Conversational agents operate on a spectrum between two different domains: *Open domain* (no clear responsibility/ purpose [3], open conversations) and *Closed domain* (specific task to fulfil, limited set of options). Depending on the domain in which the conversational agent operates, certain methods are better suited than others. The approaches we are going to describe in this paper are the *Rule-based approaches*, the *Probabilistic approaches* and the *Data driven approaches*.

In the following sections we are going to give a quick overview over the different types of context, the different domains of operation and the respective approaches used in those domains.

II. Types of context

A. World knowledge

World knowledge is the knowledge which applies to all to all users in all conversations independently of the individual preferences. This means things like time, location, weather etc. should be considered. Because what good is the conversational agent if he only proposes restaurants that are in Australia, when the user is living in Switzerland? Or if the user asks about the local time, he gets a random time.

The agent should also be aware of concepts. E.g. He should know what a gory movie means in order to be able to provide the user with an appropriate answer [4].

B. User knowledge

The user knowledge is the knowledge which a conversational agent needs to have about the specific user in order to be able to answer a query. Like the world knowledge it is also tracked over all conversations. However it is specific to the user and is learned over the course of multiple conversations. E.g. the conversational agent should know the user's personal preferences, relationships, schedules etc.

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 [1].

C. 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 a specific conversation. For example:

U1: Person: I would like to see the best Italian restaurant
U2: Conversational Agent: Hey, it is "Luigi's 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 agent 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_t 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_t 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 can take multiple actions. The user may state his Intent and/or give additional information (e.g. I want to ride the bus from $\rightarrow$ 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 [5] between which a conversational agent operates:

- **Closed domain**: The agent 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 [4] (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 conversational agent 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 kind of conversational agents are currently on the rise in the form of virtual private assistants (e.g. google assistant, amazon alexa etc.) [3]. With those kind of agents it is impossible to model a set of possible states 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 conversational agents fall somewhere along this axis (e.g. a help desk agent. It is not operating in a completely closed domain, as it has to offer solutions for a broad set of problems). 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. For open domains 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 [2]. A fixed set of heuristics decides what to do in which situation. The given input is always matched to **one** specific output state.

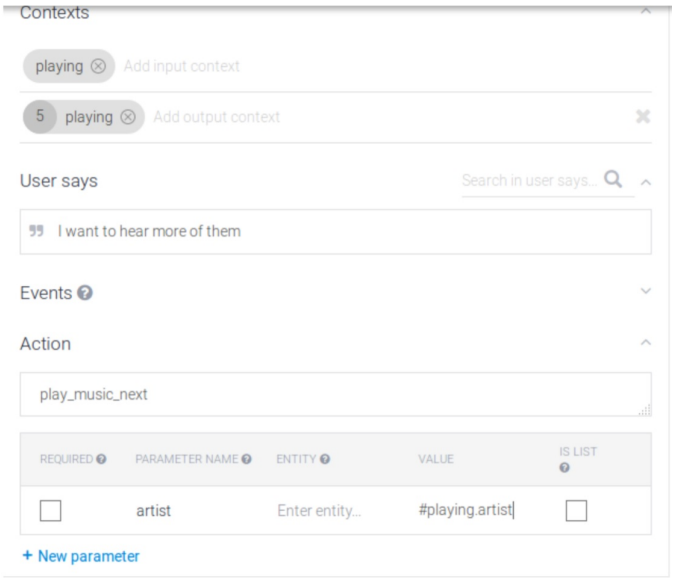

Fig. 2: Example of a rule-based approach from www.dialogflow.com

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 states is infinite 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 [6].

B. Probabilistic approaches

One of the most common and most successful approaches in this field is using a partially observable markov decision process (POMDP) [7], [8] dialogue state model. The most successful is the hidden information state (HIS) approach to dialogue management [9], [10], which is based on a POMDP.

1) POMDP: The partially observable markov decision process is described by the following:

- a set of states $S = \{s_1, s_2, ...s_{|S|}\}$
- a set of actions $A = \{a_1, a_2, ...a_{|A|}\}$
- a set of transition probabilities $T(s_i, a, s_j) = p(s_j|s_i, a)$
- a set of observation probabilities $O(z_i, a, s_j) = p(z_i|s_j, a)$
- a set of rewards $R : S \times A \rightarrow \mathbb{R}$
- a discount factor $\gamma \in [0, 1]$
- an initial belief $p_0(s)$

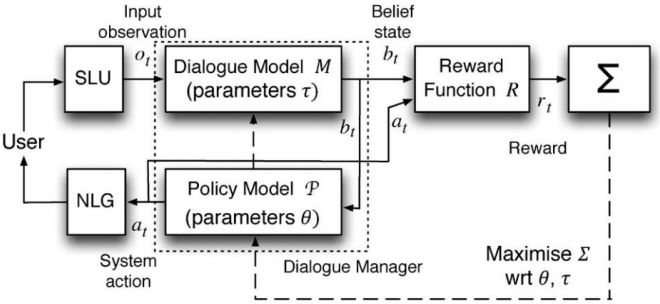

Fig. 3: Partially observable markov decision process (POMDP) based spoken dialogue system [8]

Because the state s_t is not observable and thus is not known exactly, a distribution over all possible states S is kept. This distribution is called the belief state. So the probability of being in state s_t, given the belief state **b** is b(s_t) [9]. Based on the current belief state **b** an action a_t is selected and the system receives a reward R and transitions to a new state s'_t, which is also unobserved. The system then receives an observation o' which depends on s'_t and a_t. After this the belief **b** is updated [8], [9].

The HIS model refines the way the belief monitoring works. Only a N-best list of beliefs is kept. There is also always a "Rest" value kept in the N-best list (see Figure 4) [9].

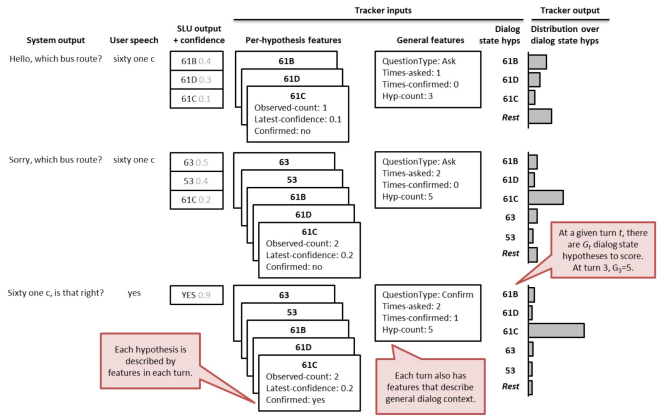

Fig. 4: Example of a probabilistic dialogue state tracking process [2]

In Figure 4 we can see a nice example of how a probabilistic model works. The example is taken from the dialogue state tracking challenge [2]. We have given a set of states and their possible output hypotheses. After the user gives the input, the most probable hypothesis is chosen based on the given start probabilities. In the example after the first input, no decision can be made. So the system asks the user for clarification. The probabilities get updated based on the last input and now the correct answer can be given to the user with the updated probabilities.

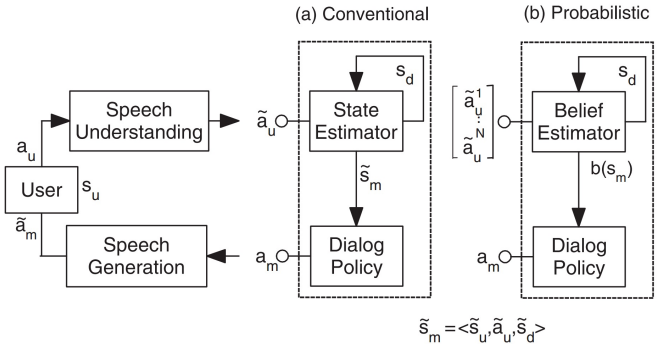

Fig. 5: Conventional model vs. probabilistic POMDP model [10]

In Figure 5 we can see two different structures of a spoken dialogue system. In (a) we see a conventional rule-based one and in (b) a probabilistic one. a_u and a_m denote user and machine dialogue acts, s_u is the user goal and s_d is the dialogue history. The tilde indicates an estimate. The conventional dialogue manager in part (a) only holds the estimate of one single state, whereas the probabilistic manager in part (b) represents a dialogue manager which maintains a distribution over all states and accepts an N-best list of alternative user inputs [10].

C. Data driven approaches

In a open domain it quickly becomes impossible to define all possible states and the corresponding transition matrix. It

is therefore not possible to handcraft a system which solves the context state tracking problem [4].

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 hand crafting a the mapping from context to response, we can learn this mapping from the data.

In general there are two conrasting data driven approaches; *Discriminative and Generative*. Most production systems today use the disctiminative approach, but the research focus is mostly on the generative approach. For both approaches there are statistical and deep learning methods available [5].

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 basically reduces the task to a retrieval problem akin to search engines (retrieving results based on a query).

Most discriminative methods use the data to learn a ranking function. This function is then used to score all possible responses. The best scoring response is given as an answer to the user. [5]

This methods have the advantage that they do not make grammatical or semantical errors (because the predefined responses were written by humans). But they can't handle unseen cases or refer back to contextual entities (e.g. names).

An example for a statistical method ist *Okapi BM25* [11]. A search engine ranking function based on "bag-of-words" retrieval.

Is is also possible to train an artificial neural network to score replies [5], [12]. This method is further explained in subsection V-C, as we further investigate this method in our ablation study.

2) Generative methods: Generative methods do not use predefined responses. They generate new responses from scratch. To do this they use approaches from machine translation. But instead of translating form one language to another they translate from context to response.

This has the benefit that it is possible to refer back to entities in the input. But these methods often make grammatical mistakes and are hard to train. They tend to default to "save" answers like "I don't know" or even "I love you".

There are some methods from statistical machine translation, which basicly try to match the phrase probability distribution in both languages. But the research focus lies on deep learning methods like the RNN Encoder-Decoder Network [13]. It uses one network to encode the context into a vector and another to decode the network into the response.

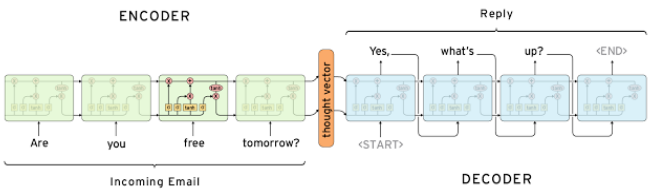

Fig. 6: The dual encoder model form the sequence to sequence paper [13]

V. Ablation Study

A. Goal

B. Dataset

We use the *Ubuntu Dialog Corpus* [14]. It is one of the largest public dialogue datasets available. Its based on chat logs from the Ubuntu channels on a public IRC network.

The training data consists of one million examples, 50% positive and 50% negative. Each example consists of a context (the conversation up to this point) and an utterance (the response to the context). A positive label means that an utterance was an actual response to a context, and a negative label means that the utterance wasnt it was picked randomly from somewhere in the corpus. [15]

The format of the testing data is slightly different from the training data. It consists of a context, a ground truth response and 9 distracting responses. The goal of the network is to pick the ground truth response from the distractors.

C. Model architecture

We use a model called *Dual Encoder LSTM* [14]. It has been reported to achieve good accuracy for this dataset [12]. This model takes as input a context and a candidate response and outputs a score indicating how good this answer fits to the context.

To do this it uses two recurrent neural networks (RNNs) - one for the context and one for the response - to encode the input into two vectors c and r. These vectors should capture the meaning and all important information of the context and response.

The context vector c is then multiplied with a learned matrix M to predict a response $c^T M = r_p$.

To calculate the score for the response we measure the similarity between the predicted response r_p and the candidate response r by taking their dot product (and applying a sigmoid to convert it to a probability). A large dot product means the network thinks that the response is a good choice for that context.

The network is trained with *binary cross-entropy loss*.

D. Methodology

Our original idea was to train two networks. One with and one without context information (only the latest utterance). And then evaluate on the same testing dataset. But in this case the one without context information would receive a different kind of input while testing than during training. We know from experience that neural networks perform very poorly when

they are faced with data that is not in the same format as the training data. Therefore the second network would have surely performed worse, falsifying the results.

To solve this we came up with the idea to instead add noise to the context. This has the added benefit that we not only have the extreme case of *with or without context* but also everything in between.

We add noise by replacing every utterance in the context (except the last one) with a different utterance based on the probability P.

We generate ten training sets with P form 0.0 to 1.0 and train one network on each set. All ten resulting networks are evaluated on the same testing dataset.

E. Results

Our results are summarised in Figure 7, Figure 8 and Table I. We observe a steady accuracy decline with rising noise. At first the difference is only small but with larger noise probability the accuracy drops drasticly. With these results it is save to say that the network actually uses the context information to implicitly model the dialogue state.

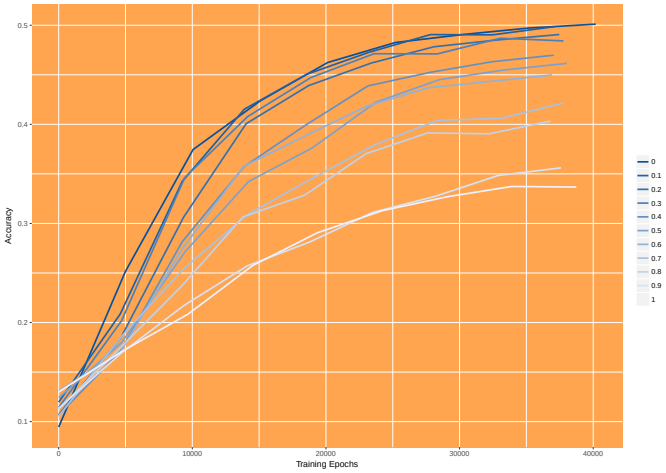

Fig. 7: Accuracy during training.

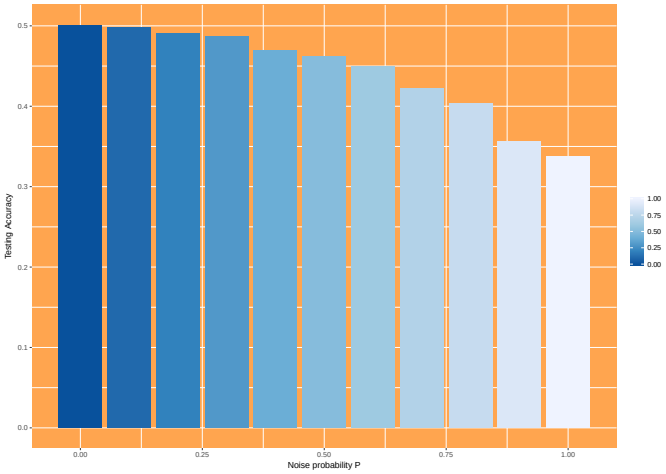

Fig. 8: Accuracy for different noising probabilies

TABLE I: Accuracy for different noising probabilies

Noise probability	ACC
0.0 (Baseline)	0.501
0.1	0.498
0.2	0.491
0.3	0.486
0.4	0.469
0.5	0.461
0.6	0.449
0.7	0.421
0.8	0.403
0.9	0.356
1.0	0.337

VI. Conclusion

Overall we can say that probabilistic models are very well suited to keep track of the dialogue context in conversational agents that operate in a more closed domain. However with a more open domain, data driven approaches are better suited. With the increasing amount of available data, those approaches become steadily more valid and more accurate.

In our ablation study we found that removing (noising) the context from the training data does decrease the performance of the network. This shows that neural networks do indeed learn to implicitly model the dialogue state and context.

References

[1] 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.

[2] J. Williams, A. Raux, D. Ramachandran, and A. Black, "The dialog state tracking challenge," in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 404–413.

[3] L. C. Klopfenstein, S. Delpriori, S. Malatini, and A. Bogliolo, "The rise of bots: A survey of conversational interfaces, patterns, and paradigms," in *Proceedings of the 2017 Conference on Designing Interactive Systems*. ACM, 2017, pp. 555–565.

[4] 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.

[5] 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.

[6] R. S. Wallace, "The anatomy of alice," in *Parsing the Turing Test*. Springer, 2009, pp. 181–210.

[7] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[8] S. Young, M. Gašić, B. Thomson, and J. D. Williams, "Pomdp-based statistical spoken dialog systems: A review," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1160–1179, 2013.

[9] S. Young, J. Schatzmann, K. Weilhammer, and H. Ye, "The hidden information state approach to dialog management," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–149.

[10] S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The hidden information state model: A practical framework for pomdp-based spoken dialogue management," *Computer Speech & Language*, vol. 24, no. 2, pp. 150–174, 2010.

[11] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1, no. 1.

[12] R. Kadlec, M. Schmid, and J. Kleindienst, "Improved deep learning baselines for ubuntu corpus dialogs," *arXiv preprint arXiv:1510.03753*, 2015.

[13] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[14] R. Lowe, N. Pow, I. Serban, and J. Pineau, "The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems," *arXiv preprint arXiv:1506.08909*, 2015.

[15] "WILDML deep learning for chatbots, part 2 implementing a retrieval-based model in tensorflow," http://www.wildml.com/2016/07/deep-learning-for-chatbots-2-retrieval-based-model-tensorflow/#more-771, accessed: 2018-02-11.