Frames: A Corpus for Adding Memory to Goal-Oriented Dialogue Systems

Layla El Asri and **Hannes Schulz** and **Shikhar Sharma** and **Jeremie Zumer**

Justin Harris and **Emery Fine** and **Rahul Mehrotra** and **Kaheer Suleman**
Microsoft Maluuba
first.last@microsoft.com

Abstract

This paper proposes a new dataset, *Frames*, composed of 1369 human-human dialogues with an average of 15 turns per dialogue. This corpus contains goal-oriented dialogues between users who are given some constraints to book a trip and assistants who search a database to find appropriate trips. The users exhibit complex decision-making behaviour which involve comparing trips, exploring different options, and selecting among the trips that were discussed during the dialogue. To drive research on dialogue systems towards handling such behaviour, we have annotated and released the dataset and we propose in this paper a task called *frame tracking*. This task consists of keeping track of different semantic frames throughout each dialogue. We propose a rule-based baseline and analyse the frame tracking task through this baseline.

1 Introduction

Goal-oriented, information-retrieving dialogue systems have been designed traditionally to help users find items in a database given a set of constraints (Singh et al., 2002; Raux et al., 2003; El Asri et al., 2014; Laroche et al., 2011). For instance, the *LET'S GO* dialogue system finds a bus schedule given a bus number and a location (Raux et al., 2003).

Available resources for data-driven learning of such goal-oriented systems are often collected with an existing system (Henderson et al., 2014b; Bennett and Rudnicky, 2002) and have been proposed to study one component of dialogue. Examples are the first three Dialogue State Tracking Challenges (DSTC, Williams et al., 2016) during which a series of datasets and tasks of increasing complexity were released. These shared tasks were essential to advance the state of the art on state tracking. Other resources have allowed to study and develop different approaches to spoken language understanding and entity extraction (Mesnil et al., 2013). As for dialogue management, simulators have been proposed (Schatzmann et al., 2006) but datasets are scarce.

In most datasets collected with an existing system, the dialogues consist of sequential slot-filling: the system requests constraints until it can query the database and return several results to the user. Then, the user can ask for more information about a given result or request other possibilities. As a consequence, the tasks and methods that were based on these datasets were defined according to this sequential slot-filling process

We propose the *Frames* dataset to study more complex dialogue flows and decision-making behaviour. Our motivation comes from user studies in e-commerce which show that several information-seeking behaviours are exhibited by users who may come with a very well defined item in mind, but may also visit an e-commerce website with the intent to compare items and explore different possibilities (Moe and Fader, 2001; Saha et al., 2017). Supporting this kind of decision-making process in conversational systems implies adding *memory*. Memory is necessary to track different items or preferences set by the user during the dialogue. For instance, consider product comparisons. If a user wants to compare different items using a dialogue system, then this system should be able to separately recall properties pertaining to each item.

We collected 1369 human-human dialogues in a Wizard-of-Oz (WOz) setting – *i.e.*, users were paired up with humans, whom we refer to as *wizards*, who assumed the role of the dialogue system. Wizards were given access to a database of vaca-

Proceedings of the SIGDIAL 2017 Conference, pages 207–219,
Saarbrücken, Germany, 15-17 August 2017.

tion packages containing round-trip flights and a hotel. Users were tasked with finding packages based on a few constraints such as a destination and a budget. The dataset has been fully annotated by human experts and is publicly available[1].

Along with this dataset, we formalize a new task called *frame tracking*. Frame tracking is an extension of state tracking (Henderson, 2015; Williams et al., 2016). In state tracking, the information summarizing the full dialogue history is compressed into a single semantic frame which contains properties and values corresponding to the user's preferences (*e.g.*, destination city). In frame tracking, the dialogue agent must simultaneously track multiple semantic frames (*e.g.*, different destination cities; frames are defined formally in Section 4.2) throughout the conversation.

2 Data Collection

We collected the Frames data over a period of 20 days with 12 participants, who worked either for one day, one week, or 20 days. The participants alternated between the user and wizard roles on a daily basis. Due to this rotation, we can assume that we deal with returning users who know how to use the system, and focus on the decision making process, skipping the phase where the user learns about the system capabilities. The domain for all dialogues is travel: specifically, finding a vacation package that fulfils certain *a priori* requirements through a conversational search-and-compare process.

2.1 Wizard-Of-Oz Setting

Wizard-of-Oz (WOz) dialogues (Kelley, 1984; Rieser et al., 2005; Wen et al., 2016) have the considerable advantage of exhibiting realistic behaviours often beyond the capabilities of existing dialogue systems. Our setting is slightly different from the usual WOz setting because, in our case, users did not believe they were interacting with a dialogue system; they knew they were conversing with fellow humans. We chose not to give templated answers to wizards because, apart from studying decision-making, we also wanted to study information presentation and dialogue management. We work with text-based dialogues because this engenders a more controlled wizard behaviour, obviates handling time-sensitive turn taking, and speech recognition noise.

[1] datasets.maluuba.com/Frames

2.2 Task Templates and Instructions

User-wizard dialogues took place on Slack.[2] We deployed a Slack bot to pair up participants and record conversations. At the beginning of each dialogue, a user was paired with a wizard and given a new task. Tasks were built from templates like the following:

> "Find a vacation between `[START_DATE]` and `[END_DATE]` for `[NUM_ADULTS]` adults and `[NUM_CHILDREN]` kids. You leave from `[ORIGIN_CITY]`. You are travelling on a budget and you would like to spend at most $`[BUDGET]`."

Tasks were generated by drawing values (*e.g.*, for `BUDGET`) from a database. We constructed our database of flight and hotel properties by hand to simulate what one would find on a standard travel booking site. Each template was assigned a probability of success, and then constraint values were drawn in order to comply with this probability. For example, if 20 tasks were generated at probability 0.5, about 10 tasks would be generated with successful database queries and the other 10 would be generated such that the database returned no results for the constraints. This success mechanism allowed us to emulate cases when a user would find nothing meeting her constraints. If a task was unsuccessful, the user either ended the dialogue or got an alternative task such as: "If nothing matches your constraints, try increasing your budget by $200." We wrote 38 templates. 14 were generic like the one presented above and the other 24 included a background story to encourage role-playing from users and to keep them engaged. These templates were meant to add variety to the dialogues. The generic templates were also important for the users to create their own character and personality. We found that the combination of the two types of templates prevented the task from becoming too repetitive. Notably, we distributed the role-playing templates throughout the data collection process to bring some novelty and surprise. We also asked the participants to write templates (13 of them) to keep them engaged in the task.

To control data collection, we gave a set of instructions to the participants. The user instructions encouraged a variety of behaviours. As for the wizards, they were asked only to talk about the

[2] www.slack.com

database results and the task at hand. We also asked the wizards to perform untimely actions occasionally, for instance, to ask for information that the user has already provided. It is interesting from a dialogue management point of view to have examples of bad behaviour and of how it impacts user satisfaction. At the end of each dialogue, the user provided a wizard cooperativity rating on a scale of 1 to 5. The wizard, on the other hand, was shown the user's task and was asked whether she thought the user had accomplished it.

2.3 Search Interface And Suggestions

Wizards received a link to a search interface every time a user was connected to them. The search interface was a simple GUI with all the searchable fields in the database (see Appendix A). For every database search, up to 10 results were displayed, sorted by increasing price.

Another important property of human dialogue that we want to study with Frames is how to provide users with database information. When a set of user constraints leads to no results, users would benefit from knowing that relaxing a given constraint (*e.g.*, increasing the budget by a reasonable amount) leads to results. We modelled this by displaying suggestions to the wizards when a database query returned no results. Suggestions were packages obtained by randomly relaxing one or more constraints. It was up to the wizard to decide whether or not to use suggestions.

3 Statistics of the Corpus

Using the data collection process described above, we collected *1369 dialogues*. Figure 1a shows the distribution of dialogue lengths in the corpus. The average number of turns is 15, for a total of *19986 turns* in the dataset. A turn is defined as a Slack message sent by either a user or a wizard. Turns always alternate between user and wizard.

Figure 1b shows the number of acts per dialogue turn. About 25% of the dialogue turns have more than one dialogue act. The turns without dialogue acts are turns where the user asked for something that the wizard could not provide, *e.g.*, because it was not part of the database. We left such (rarely occurring) user turns unannotated, as they are usually followed up by the wizard saying she cannot provide the required information. This rarely occurs, since our users are familiar with the capabilities of the "system" after only few dialogues.

Figure 1c shows the distribution of user ratings. More than 70% of the dialogues have the maximum rating of 5. Figure 2 shows the occurrences of dialogue acts in the corpus. The dialogue acts are described in Table 9. We present the annotation scheme in the following section.

4 Annotation

We manually annotated the Frames dataset with dialogue acts, slot types and values, references to other frames, and the ID of the currently active frame for each utterance. We also computed frame descriptions based on the labels of earlier turns.

4.1 Dialogue Acts, Slot Types, Slot Values

Most of the dialogue acts used for annotation are typical of the goal-oriented setting, such as `inform` and `offer` (Henderson et al., 2014b). We also introduced dialogue acts specifically for frame tracking, such as `switch_frame` and `request_compare`. The dialogue acts are listed in Table 9.

Our annotation uses three sets of slot types. The first set, listed in Tables 7 and 8, corresponds to the fields of the database. The second set is listed in Table 10 and contains the slot types which we defined to describe specific aspects of the dialogue, such as `intent`, `action`, and `count`. The remaining slot types in Table 10 were introduced to describe frames and cross-references between them.

4.2 Frame Definition

Semantic frames form the core of our dataset. A semantic frame is defined by the following four components:

- User requests: slots whose values the user wants to know for this frame.
- User binary questions: user questions with slot types and slot values.
- Constraints: slots which have been set to a particular value by the user or the wizard.
- User comparison requests: slots whose values the user wants to know for this frame and one or more other frames.

In DSTC, a semantic frame contains the constraints set by the user, the user requests, and the user's search method (*e.g.*, by constraints or alternatives). In our case, constraints can also be set by the wizard when she suggests or offers a package. Any field in the database (see Tables 7 and 8 in Appendix A) can be constrained by the user or

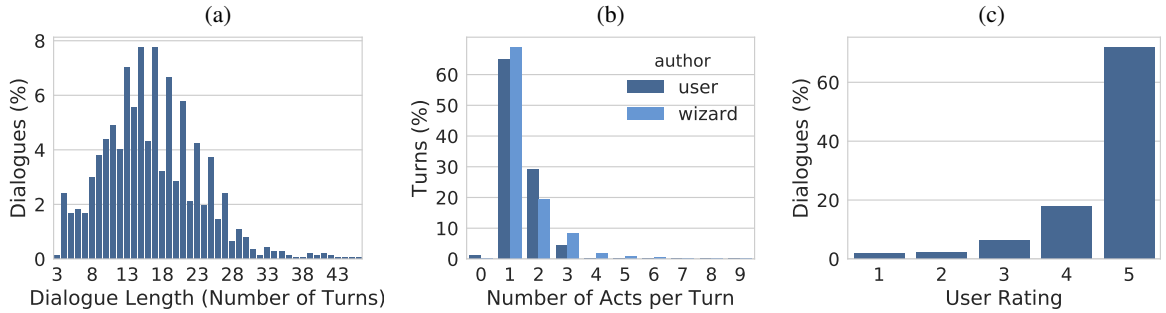

Figure 1: Overview of the Frames corpus

Table 1: Dialogue excerpt with active frame annotation

Author	Utterance	Frame
User	I'd like to book a trip to Atlantis from Caprica on Saturday, August 13, 2016 for 8 adults. I have a tight budget of 1700.	1
Wizard	Hi...I checked a few options for you, and unfortunately, we do not currently have any trips that meet this criteria. Would you like to book an alternate travel option?	1
User	Yes, how about going to Neverland from Caprica on August 13, 2016 for 5 adults. For this trip, my budget would be 1900.	2
Wizard	I checked the availability for those dates and there were no trips available. Would you like to select some alternate dates?	2

the wizard. The comparison requests and the binary questions were added after analysing the dialogues. The comparison requests correspond to the `request_compare` dialogue act. This dialogue act is used to annotate turns when a user asks to compare different results, for instance: "*Could you tell me which of these resorts offers free wifi?*". These questions possibly relate to several frames. Binary questions are questions with slot types and slot values, *e.g.*, "*In which part of the town is the hotel located?*" (`request` act), or "*Is the trip to Marseille cheaper than to Naples?*" (`request_compare` act), as well as all `confirm` acts. Binary questions may concern one or several frames.

4.3 Frame Creation and Switching

Each dialogue starts in frame 1. New frames are introduced when the wizard offers or suggests something, or when the user modifies pre-established slots. Thus, all values discussed during the dialogue are recorded and the user can return to a previous set of constraints at any point. An example is given in Table 1: the frame number changes when the user modifies several slot values, namely, the destination city, the number of adults for the trip, and the budget. While modifying pre-established slots is supported by most dialogue systems, these rules allow us to clearly distinguishing creating frames from extending frames and thus define how the items in the dialogue memory, which the user can reference, are structured. Though frames are created for each offer or suggestion made by the wizard, the *active* frame can only be changed by the user so that the user has control over the dialogue. When creating frames, the annotator can explicitly mark which frame the new frame is derived from, which heuristically copies some of its content to the new frame. If not annotated, we assume it is derived from the currently active frame. If the user asks for more information about a specific offer or suggestion, the active frame is changed to the frame introduced with that offer or suggestion. This change of frame is indicated by a `switch_frame` act (see Appendix A). The rules for creating and switching frames are summarized in Table 2.

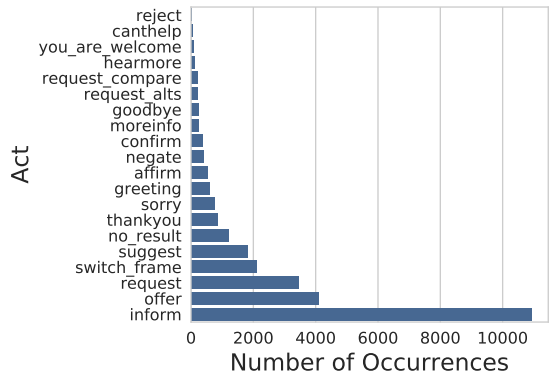

Figure 2: Dialogue act occurrences in the corpus

We introduced specific slot types for recording the creation and modification of frames. These slot types are `id`, `ref`, `read`, and `write` (see Table 10 in Appendix A). The frame `id` is defined when the frame is created and is used to switch to

Table 2: Frequency of frame creation and switching events

Rule Type	Author	Rule Description	Relative Frequency	Absolute Frequency
Creation	User	Changing the value of a slot	31 %	2092
	Wizard	Making an offer or a suggestion	69 %	4762
Switching	User	Changing the value of a slot (it causes the dialogue to switch to that frame)	50 %	2092
		Considering a wizard offer or suggestion	39 %	1635
		Switching to an earlier frame by mentioning its slot values	11 %	458

this frame when the user decides to do so.

The other slot types are used to annotate cross-references between frames. A reference has two parts: the `id` of the frame it refers to and the slots and values that are used to refer to that frame (if any). For instance, `ref[1{name=Tropic}]` means that frame 1 is being referred to by the hotel name *Tropic*. If anaphora are used to refer to a frame, we annotated this with the slot `ref_anaphora` (*e.g.*, "This is too long" – `inform(duration=too long, ref_anaphora=this)`). Inside an `offer` dialogue act, a `ref` means that the frame corresponding to the offer is derived from another frame. This happens for instance when a wizard proposes a package with business or economy options. In this case, the business and economy offers are derived from the hotel offer.

The slot types `read` and `write` only occur inside a wizard's `inform` act and are used by wizards to provide relations between offers or suggestions: `read` is used to indicate which frame the values come from (and which slots are used to refer to this frame, if any), while `write` indicates the frame where the slot values are to be written (and which slot values are used to refer to this frame, if any). If there is a `read` without a `write`, the current frame is assumed as the storage for the slot values. A slot type without a value indicates that the value is the same as in the referenced frame, but was not mentioned explicitly *e.g.*, "for the same price".

Table 3 gives an example of how these slot types are used in practice: `inform(`*`read`*`=[7{dst_city=Punta Cana, category=2.5}])` means that the values *2.5* and *Punta Cana* are to be read from frame 7, and to be written in the current frame. At this turn of the dialogue, the wizard repeats information from frame 7. The annotation `inform(breakfast=False,`*`write`*`=[7{name=El Mar}])` means that the value *False* for breakfast is written in frame 7 and that frame 7 was identified in this utterance by the name of the hotel *El Mar*.

The average number of frames created per dialogue is 6.71 and the average number of frame switches is 3.58. Figure 3 shows boxplots for the number of frame creations and the number of frame changes in the corpus.

4.4 Annotation Reproducibility

Five trained experts annotated the dataset according to the above rules. To measure inter-annotator agreement, the experts annotated the same randomly chosen 10 dialogues. On this subset, we compute the inter-annotator agreement rate as the F1-score. Note that the commonly used κ statistic cannot be directly applied here, since the annotation is not a multi-class classification problem. The provided F1 score also captures how much the annotators failed to annotate words or acts, or disagreed about the correct value. We report the mean and standard deviation over all possible pairing of annotators. On dialogue acts only, this score is 81.2 $\pm$ 3.1, on slot values, it is 95.2 $\pm$ 1.1, and on dialogue acts, slot values, and content of referenced frames, it is 62.3 $\pm$ 4.9.

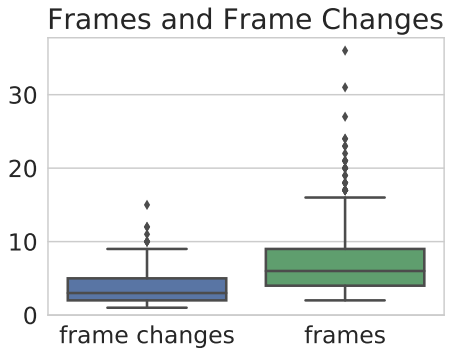

Figure 3: Number of frames and frame switches in the corpus

Table 3: Annotation example with the `write` and `read` slot types

Author	Utterance	Frame	Annotation
Wizard	I am only able to find hotels with a 2.5 star rating in Punta Cana for that time.	6	inform(*read*=[7{dst_city=Punta Cana, category=2.5}])
User	2.5 stars will do. Can you offer any additional activities?	11	inform(category=2.5)
Wizard	Unfortunately I am not able to provide this information.	11	sorry, canthelp
User	How about breakfast?	11	request(breakfast)
Wizard	El Mar does not provide breakfast.	11	inform(breakfast=False, *write*=[7{name=El Mar}])

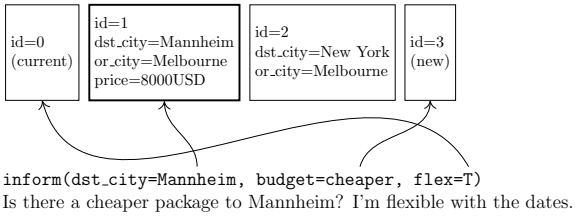

Figure 4: Illustration of the frame tracking task. The model must choose, for each slot, which frame it is referring to, given the set of available frames, the previous active frame (bold), and the potential new frame (marked "(new)").

5 Research Topics

Frames can be used to study many aspects of goal-oriented dialogue, from Natural Language Understanding (NLU) to Natural Language Generation (NLG). In this section, we propose three topics that we believe are new and representative of Frames.

5.1 Frame Tracking

5.1.1 Definition

We propose *Frame tracking* as an extension of state tracking (Henderson, 2015) to a setting where several semantic frames are tracked simultaneously. In state tracking, the dialogue history is compressed into one semantic frame. The state tracker updates a probability distribution, for each slot, over the different possible values. Every time the user sets a new value, the probability distribution is updated. This architecture prevents the user from comparing options or returning to an item discussed earlier since the values for each slot are tracked separately. In frame tracking, a new value creates a new semantic frame. The frame tracking task is significantly harder as it requires, for each user utterance, identifying the active frame as well as all the frames modified by the utterance. An example is provided in Fig. 4.

Definition 1 (Frame Tracking)**.** At each user turn t, we assume access to the full dialogue history $H = \{f_1, ..., f_{n_{t-1}}\}$, where f_i is a frame and n_{t-1} is the number of frames created so far in the dialogue. For a user utterance u_t at time t, we provide the following NLU labels: dialogue acts, slot types, and slot values. The goal of frame tracking is to predict if a new frame is created and to predict for each dialogue act the `ref` labels (possibly none) and the `ids` of the frames referenced.

Predicting the frame that is referenced by a dialogue act requires detecting if a new frame is created and recognizing a previous frame from the values mentioned by the user (potentially synonyms, *e.g.*, NYC for New York), or by using the user utterance directly. It is necessary in many cases to use the user utterance directly because users do not always use slot values to refer to previous frames. An example in the corpus is a user asking: "Which package has the soonest departure?". In this case, the user refers to several frames (the packages) without ever explicitly describing which ones. This phenomenon is quite common for dialogue acts such as `switch_frame` (979 occurrences in the corpus) and `request_compare` (455 occurrences in the corpus). These cases can only be resolved by working on the text directly and solving anaphora.

Note that when talking with real users, a system would need to generate the frames dynamically during the dialogue. We propose the frame tracking task as a first step and we show in Section 6.2 that this simplified task entails many challenges.

5.1.2 Evaluation Metrics

We define two metrics: frame identification and frame creation. For frame identification, for each dialogue act, we compare the ground truth pair (`key-value`, `frame`) to that predicted by the frame tracker. We compute performance as the number of correct predictions over the number of

pairs. The `frame` is the `id` of the referenced frame. The `key` and `value` are respectively the type and the value of the slot used to refer to the frame (these can be null).

For frame creation, we compute the number of times the frame tracker correctly predicts that a frame is created or correctly predicts that a frame has not been created over the number of dialogue turns.

5.1.3 Related Work

In previous work, some limitations of sequential slot filling dialogue systems were addressed using goal-modeling (Crook and Lemon, 2010; Crook et al., 2012; Misu et al., 2011), task tracking (Lee and Stent, 2016) and memory-augmentation of classical state tracking (Weston et al., 2015).

Crook and Lemon (2010); Crook et al. (2012) model the user goal as a subset of all possible slot value combinations and propose techniques to automatically compress this huge space into a summary space. Rewards, transitions, and observations of a POMDP system can then be projected to the reduced space, which facilitates policy learning. Misu et al. (2011) propose a method for decision support in spoken dialogue systems that aids a user who is assumed to have an (unknown) weighted preference over the possible slot values and limited knowledge about alternatives. The authors employ a user simulator that outputs dialogue acts to learn a policy that optimizes the sum of the weights of the final user selection. The Frames dataset allows learning and evaluating these techniques on a large and more realistic text-based dataset. Additionally, the memorized frames would allow a dialogue system to compare disjunct goals or return to earlier states.

Recent approaches to state tracking have been suggested to go beyond the sequential slot-filling approach. An important contribution is the Task Lineage-based Dialog State Tracking (TL-DST) proposed by Lee and Stent (2016). TL-DST is a framework that allows keeping track of tasks across different domains. Similarly to frame tracking, Lee and Stent propose building a dynamic structure of the dialogue containing different frames corresponding to different tasks. They defined different sub-tasks among which *task frame parsing* which is closely related to frame tracking except that they impose constraints on how a dialogue act can be assigned to a frame and a dialogue act can only relate to one frame. Because of the lack of data, Lee and Stent (2016) trained their tracking model on datasets released for DSTC (DSTC2 and DSTC3, Henderson et al., 2014b,a). As a result, they could artificially mix different tasks, *e.g.*, looking for a restaurant and looking for a pub, but they could not study how human beings switch between topics. In addition, this framework can switch between different tasks but does not handle comparisons between disjunct frames, which is an important aspect of frame tracking.

Another related approach was proposed by Perez and Liu (2016) who re-interpreted the state tracking task as a question-answering task. Their state tracker is based on a memory network (Weston et al., 2015) and can answer questions about the user goal at the end of the dialogue. They also propose adding functionalities such as keeping a list of the constraints expressed by the user during the dialogue.

The Frames dataset may be used to test and validate these approaches on real data. In addition, we propose the frame tracking task as benchmark and as a first step towards modelling complex decision-making behaviour.

5.2 Dialogue Management

Most of the time, the wizard would speak about the current frame to ask or answer questions. However, sometimes, the wizard would talk about previous frames. An example is given in Table 11 in Appendix A. In the bold utterance in this dialogue, the wizard mentions a frame which is not the currently active frame. In order to reproduce this kind of behaviour, a dialogue manager would need to be able to identify potentially relevant frames for the current turn and to output actions for these frames.

Table 11 also illustrates another novelty. In the utterance in italics, the wizard actually performs two actions. The first action consists of informing the user about the price of the *regal resort* and the second action consists of proposing another option, *Hotel Globetrotter*. Performing more than one action per turn is a challenge when using reinforcement learning (Pietquin et al., 2011; Gašić et al., 2012; Fatemi et al., 2016) and, to our knowledge, has only been tackled in a simulated setting (Laroche et al., 2009).

5.3 Natural Language Generation

An interesting behaviour observed in Frames is that wizards often tend to summarize database results. An example is a wizard saying: *"The cheapest*

Table 4: F1 scores for the NLU baseline (mean and standard deviation).

Dialogue Acts	Slots
0.78 ± 0.05	0.79 ± 0.04

available flight is 1947.14USD." In this case, the wizard informs the user that the database has no cheaper result than the one she is proposing. To imitate this behaviour, a natural language generator (Oh and Rudnicky, 2000; Wen et al., 2015; Sharma et al., 2017) would need to reason over the database and decide how to tailor the results to the user and present them in a concise but sufficient way. Various strategies and their combinations can be employed, e.g. summarization, comparison or recommendation (Rieser and Lemon, 2009). A decision-theoretical foundation of such an approach was presented by Walker et al. (2004). A data-driven approach to attribute selection for NLG as planning under uncertainty was proposed by Rieser et al. (2014). The Frames dataset contains a larger set of dialogues as well as wizard-generated text with detailed annotations, which we believe will provide insight into when humans use which strategy and how they present the information.

6 Baselines

We developed baseline models for natural language understanding and frame tracking.

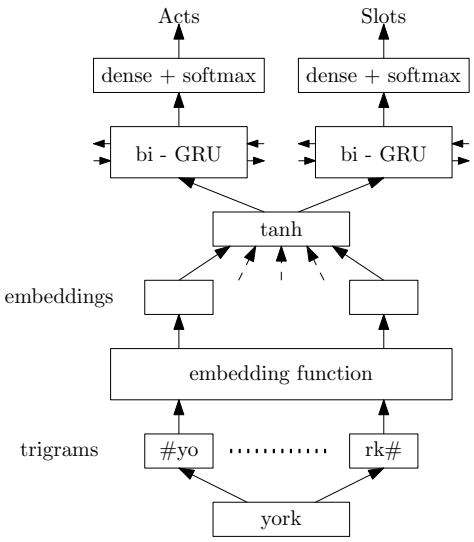

Figure 5: Illustration of the NLU model for slots and acts prediction.

Table 5: Accuracy (%) of the Frame Tracking Baselines (mean and standard deviation).

	Rule-Based	Random
Frame Creation	0.49 ± 0.03	0.47 ± 0.02
Frame Identification	0.24 ± 0.02	0.18 ± 0.02

6.1 Natural Language Understanding

We define the NLU task as dialogue act prediction and IOB (Inside, Outside, Beginning) tagging. The NLU model that we propose as baseline is illustrated in Fig. 5. We predict, for each word of the utterance, a pair of tags – one for the act and one for the slot. This model operates on character trigrams and is based on the robust named entity recognition model (Arnold et al., 2016) except that it has two heads instead of one: one head for the slot type (either a slot type or an *O* tag) as in the original model and one head for dialogue act prediction. These two parts share an embedding matrix for the input character trigrams.

We generated the IOB tags by matching the slot values in the manual annotations with the corresponding textual utterances. Note that the model only predicts IOB tags for slots whose values can be found in the text. Therefore, the prediction for slots such as `intent` or vicinities and amenities is not evaluated for this simple baseline. The act tags were also generated at the word level: for a given dialogue act with slot values, each word between the slot value that occurred first in the text and the one that occurred last in the text was tagged with the corresponding act. For example, for the utterance *I am only able to find hotels with a 2.5 star rating in Punta Cana for that time.*, the words *2.5 star rating in Punta Cana* are tagged with the `inform` dialogue act. The other words are tagged with *O*.

The two parts of the model are trained simultaneously, using a modified categorical crossentropy loss for both sets of outputs. We modify the loss to ignore *O* labels that are already predicted correctly by the model. We introduce this modification because *O* labels are far more frequent than other labels, and not limiting their contribution to the loss causes the model to degenerate to predicting *O* labels for every word. The losses for both parts of the model are added together and the combined objective is optimized using ADAM (Kingma and Ba, 2015).

We provide F1 scores for acts and slots for this model in Table 4. We report average and stan-

dard deviation over ten leave-one-user-out splits of the Frames dataset. We had a total of 11 participants who played the user role at least once during data collection. Two participants performed significantly fewer dialogues than the others. We merged the dialogues generated by these two participants (ids U21E41CQP and U23KPC9QV). For each of the resulting 10 users, we randomly split the combined dialogues of the nine others into training (80%) and validation (20%), and then tested on the dialogues from the held-out user.

6.2 Frame Tracking

We propose a rule-based frame tracking baseline which takes as input the dialogue acts with slot types and slot values but without the referenced frames (*i.e.*, the `ref` slots) as well as all the frames created so far during the dialogue. Based on this input, the tracker predicts the `ref` tags (for frame identification, see Section 5.1.2) for each dialogue act, and it predicts if a frame is created. We write $f[k]$ to denote the value of slot k in frame f. For an act $a(k{=}v)$ in frame f, the following rules are used:

- *Create and switch to a new frame* if $f[k]$ is set and a is `inform`, but v does not match $f[k]$.
- *Switch to frame* g if a is `switch_frame` and $g[k]$ matches v. If no match is found, switch to the most recently created frame.[3]
- *Assign* `ref` *to frame* g if a can have a `ref` tag, and $g[k]$ matches v. The most recently created frame is used in ambiguous cases. If no match is found, assign `ref` to the current frame.

We compare this baseline to random performance. For random performance, for each (dialogue act, slot type) combination, we compute priors on the corpus for each time the user would refer to the current frame *vs* a previous one. We sampled whether each slot referred to the current frame or another one based on that prior, and if it referred to another frame, the frame number for that other frame was sampled uniformly from the list of frames created so far.

Table 5 presents results for these baselines. We report results over 10 runs following the same evaluation method as for the NLU model. Table 5 shows that the rule-based model performs only slightly better than random on frame identification and performs similarly on frame creation. Table 6 presents an analysis of the performance of the rule-based model. We report the accuracy of the frame tracking baseline on the most crucial sub-tasks of frame tracking for one fold. The top table shows that the most difficult tasks consist of assigning the correct frame to a `switch_frame` act when the act is not directly preceded by an offer and when the act has no slots. As discussed previously, when the act has no slots, it is important to consider the text and solve anaphora. When the act is directly preceded by an offer, the baseline assigns the previous frame, which is the frame of the offer and which most of the time is the frame that the user switched to, *e.g.*, to ask for more information about the offer. In terms of frame creation, the baseline has very poor performance in correctly predicting that a frame is created because the user changes the value of a previously set slot. These results demonstrate that frame tracking cannot be solved with simple rules and necessitates tackling many complex sub-tasks.

[3] a reasonable assumption since this case often happens when a wizard makes an offer and the user talks about it.

Table 6: Accuracy (%) of the rule-based baseline on sub-tasks of frame tracking.

	With slots	Without slots	After an offer	Not after an offer
Frame switching	44.9	16.3	54.3	16.5

	Change of value	No frame creation
Frame creation	5.5	83.1

7 Conclusion

We introduced the Frames dataset: a corpus of human-human dialogues in a travel domain. This dataset contains complex user behaviour such as comparing between offers. We formalized the frame tracking task, which requires tracking simultaneously several semantic frames during a dialogue. We proposed a rule-based model for this task and analysed its performance. We release Frames in the hope of driving further research on complex decision-making in the dialogue community.

References

S. Arnold, F. A. Gers, T. Kilias, and A. Löser. 2016. Robust named entity recognition in idiosyncratic domains. *arXiv:1608.06757 [cs.CL]* .

Christina L. Bennett and Alexander I. Rudnicky. 2002. The carnegie mellon communicator corpus. In *Proc. of Interspeech*.

Paul A Crook and Oliver Lemon. 2010. Representing uncertainty about complex user goals in statistical dialogue systems. In *Proc. of SIGDIAL*. pages 209–212.

Paul A Crook, Zhuoran Wang, Xingkun Liu, and Oliver Lemon. 2012. A statistical spoken dialogue system using complex user goals and value directed compression. In *Proc. of EACL*. Association for Computational Linguistics, pages 46–50.

Layla El Asri, Rémi Lemonnier, Romain Laroche, Olivier Pietquin, and Hatim Khouzaimi. 2014. NASTIA: Negotiating Appointment Setting Interface. In *Proc. of LREC*.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. In *Proc. of SIGDIAL*.

M. Gašić, M. Henderson, B. Thomson, P. Tsiakoulis, and S. Young. 2012. Policy optimisation of POMDP-based dialogue systems without state space compression. In *Proc. of SLT*.

M. Henderson, B. Thomson, and J. Williams. 2014a. The Third Dialog State Tracking Challenge. In *Proceedings of IEEE Spoken Language Technology*.

Matthew Henderson. 2015. Machine learning for dialog state tracking: A review. In *First International Workshop on Machine Learning in Spoken Language Processing*.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014b. The second dialog state tracking challenge. In *Proc. of SIGDIAL*.

John F. Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transaction on Information Systems* .

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *Proc. of ICLR* .

Romain Laroche, Ghislain Putois, Philippe Bretier, Martin Aranguren, Julia Velkovska, Helen Hastie, Simon Keizer, Kai Yu, Filip Jurčíček, Oliver Lemon, and Steve Young. 2011. Report D6.4 : Final evaluation of classic towninfo and appointment scheduling systems. Technical report, CLASSIC Project.

Romain Laroche, Ghislain Putois, Philippe Bretier, and Bernadette Bouchon-Meunier. 2009. Hybridisation of expertise and reinforcement learning in dialogue systems. In *Proc. of Interspeech*.

Sungjin Lee and Amanda Stent. 2016. Task lineages: Dialog state tracking for flexible interaction. In *Proc. of SIGDIAL*.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Proc. of Interspeech*.

Teruhisa Misu, Komei Sugiura, Tatsuya Kawahara, Kiyonori Ohtake, Chiori Hori, Hideki Kashioka, Hisashi Kawai, and Satoshi Nakamura. 2011. Modeling spoken decision support dialogue and optimization of its dialogue strategy. *Transactions on Speech and Language Processing* 7(3):10.

Wendy W. Moe and Peter S. Fader. 2001. Uncovering patterns in cybershopping. *California Management Review* .

Alice H. Oh and Alexander I. Rudnicky. 2000. Stochastic language generation for spoken dialogue systems. In *Proc. of ANLP/NAACL workshop on Conversational Systems*.

Julien Perez and Fei Liu. 2016. Dialog state tracking, a machine reading approach using memory network. In *Proc. of EACL*.

Olivier Pietquin, Matthieu Geist, Senthilkumar Chandramohan, and Hervé Frezza-Buet. 2011. Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transaction on Speech and Language Processing* 7(3):1–21.

Antoine Raux, Brian Langner, Allan Black, and Maxine Eskenazi. 2003. LET'S GO: Improving Spoken Dialog Systems for the Elderly and Non-natives. In *Proc. of Eurospeech*.

Verena Rieser, Ivana Kruijff-Korbayov, and Oliver Lemon. 2005. A corpus collection and annotation framework for learning multimodal clarification strategies. In *Proc. of SIGDIAL*.

Verena Rieser and Oliver Lemon. 2009. Natural language generation as planning under uncertainty for spoken dialogue systems. In *Proc. of EACL*. pages 683–691.

Verena Rieser, Oliver Lemon, and Simon Keizer. 2014. Natural language generation as incremental planning under uncertainty: adaptive information presentation for statistical dialogue systems. *Transactions on Audio, Speech and Language Processing* 22(5):979–994.

A. Saha, M. Khapra, and K. Sankaranarayanan. 2017. Multimodal Dialogs (MMD): A large-scale dataset for studying multimodal domain-aware conversations. *arXiv:1704.00200 [cs.CL]* .

Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The knowledge engineering review* 21(2):97–126.

Shikhar Sharma, Jing He, Kaheer Suleman, Hannes Schulz, and Philip Bachman. 2017. Natural language generation in dialogue using lexicalized and delexicalized data. *Proc. of ICLR Workshop* .

Satinder Singh, Michael Kearns, Diane Litman, and Marilyn Walker. 2002. Optimizing dialogue management with reinforcement learning: experiments with the njfun system. *Journal of Artificial Intelligence Research* 16:105–133.

M.a. Walker, S.j. Whittaker, A. Stent, P. Maloor, J. Moore, M. Johnston, and G. Vasireddy. 2004. Generation and evaluation of user tailored responses in multimodal dialogue. *Cognitive Science* 28(5):811–840.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina Maria Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve J. Young. 2016. A network-based end-to-end trainable task-oriented dialogue system. In *Proc. of EACL*.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proc. of EMNLP*.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. *Proc. of ICLR* .

Jason D. Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue and Discourse* .

A Annotation Details and Dialogue Example

Table 7: Searchable fields in the database of packages

Field	Description
PRICE_MAX	Maximum price the user is willing to pay
PRICE_MIN	Minimum price defined by the user
DESTINATION_CITY	Destination city
MAX_DURATION	Maximum number of days for the trip
NUM_ADULTS	Number of adults
NUM_CHILDREN	Number of children
START_DATE	Start date for the trip
END_DATE	End date for the trip
ARE_DATES_FLEXIBLE	Boolean value indicating whether or not the user's dates are flexible. If True, then the search is broadened to 2 days before START_DATE and 2 days after END_DATE.
ORIGIN_CITY	Origin city

Table 8: Non-searchable fields in the database of packages

Field	Description
Global Properties	
PRICE	Price of the trip including flights and hotel
DURATION	Duration of the trip
Hotel Properties	
NAME	Name of the hotel
COUNTRY	Country where the hotel is located
CATEGORY	Rating of the hotel (in number of stars)
CITY	City where the hotel is located
GUEST_RATING	Rating of the hotel by guests (in number of stars)
BREAKFAST, PARKING, WIFI, GYM, SPA	Boolean value indicating whether or not the hotel offers this amenity.
PARK, MUSEUM, BEACH, SHOPPING, MARKET, AIRPORT, UNIVERSITY, MALL, CATHEDRAL, DOWNTOWN, PALACE, THEATRE	Boolean value indicating whether or not the hotel is in the vicinity of one of these.
Flights Properties	
SEAT	Seat type (economy or business)
DEPARTURE_DATE_DEP	Date of departure to destination
DEPARTURE_DATE_ARR	Date of return flight
DEPARTURE_TIME_DEP	Time of departure to destination
ARRIVAL_TIME_DEP	Time of arrival to destination
DEPARTURE_TIME_ARR	Time of departure from destination
ARRIVAL_TIME_ARR	Time of arrival to origin city
DURATION_DEP	Duration of flight to destination
DURATION_ARR	Duration of return flight

Table 9: List of dialogue acts in the annotation of Frames

Dialogue Act	Speaker	Description
inform	User/Wizard	Inform a slot value
offer	Wizard	Offer a package to the user
request	User/Wizard	Ask for the value of a particular slot
switch_frame	User	Switch to a frame
suggest	Wizard	Suggest a slot value or package that does not match the user's constraints
no_result	Wizard	Tell the user that the database returned no results
thankyou	User/Wizard	Thank the other speaker
sorry	Wizard	Apologize to the user
greeting	User/Wizard	Greet the other speaker
affirm	User/Wizard	Affirm something said by the other speaker
negate	User/Wizard	Negate something said by the other speaker
confirm	User/Wizard	Ask the other speaker to confirm a given slot value
moreinfo	User	Ask for more information on a given set of results
goodbye	User/Wizard	Say goodbye to the other speaker
request_alts	User	Ask for other possibilities
request_compare	User	Ask the wizard to compare packages
hearmore	Wizard	Ask the user if she'd like to hear more about a given package
you_are_welcome	Wizard	Tell the user she is welcome
canthelp	Wizard	Tell the user you cannot answer her request
reject	Wizard	Tell the user you did not understand what she meant

Table 10: List of slot types not present in the database

Slot Type	Description
count	Number of different packages
count_amenities	Number of amenities
count_name	Number of different hotels
count_dst_city	Number of destination cities
count_seat	Number of seat options (for flights)
count_category	Number of star ratings
id	Id of the frame created (for offers and suggestions)
vicinity	Vicinity of the hotel
amenities	Amenities of the hotel
ref_anaphora	Words used to refer to a frame *e.g.*, "the second package'
impl_anaphora	Used when a slot type is not specifically mentionned *e.g.*, "What is the price for Rio?"..."And for Cleveland?"
ref	Id of the frame that the speaker is referring to
read	Reads slot values specified in another frame and writes them in the current frame
write	Writes slot values in a given frame
intent	User intent (*e.g.*, book)
action	Wizard action (*e.g.*, book)

Table 11: Dialogue excerpt where the wizard talks about a frame other than the active frame

Author	Utterance
Wizard	A 5 star hotel called the Regal Resort,
Wizard	it has free wifi and a spa.
User	dates?
Wizard	Starts on august 27th until the 30th
User	ok that could work. I would like to see my options in Santos as well
Wizard	*regal resort goes for $2800 or there is the Hotel* Globetrotter in Santos it has 3 stars and comes with *breakfast and wifi, it leaves on the 25th and returns on the 30th! all for $2000*
User	ahh I can't leave until august 26 though
Wizard	**then i guess you might have to choose the Regal resort**
User	yeah. I will book it
Wizard	Thank you!