Roving Mind: a balancing act between open–domain and engaging dialogue systems

A. Cervone, G. Tortoreto, S. Mezza, E. Gambi, G. Riccardi
Signals and Interactive Systems Lab., University of Trento, Italy
{alessandra.cervone,giuliano.tortoreto,stefano.mezza}@unitn.it
{enrico.gambi,giuseppe.riccardi}@unitn.it

Abstract

Open–domain dialogue systems should be able to cover a very large set of domains and at the same time keep the user engaged in the interaction. Current approaches to dialogue modeling are divided between domain–independent, non–modular approaches using sequence–to–sequence models and the domain–specific modular systems developed for task–based dialogue. Furthermore, user engagement in dialogue, addressed in this bot challenge, is a rather new research area. In this work we describe Roving Mind, a dialogue system which combines domain–independence with a modular architecture for open–domain spoken conversation, with a specific module for user engagement. Our architecture takes a balanced approach between human expert design and data–driven approaches, adapting the traditional task–based architecture relying on slots and intents to open–domain conversation using entities and domain-independent dialogue acts. The system was tested extensively during the Alexa Prize semifinals and received ratings from a large number of users, which allowed us to draw statistically relevant facts on how users interact with an open domain, non task-based social bot. Our experiments show that on average users give higher ratings to task-driven conversations (using user–entertaining strategies) compared to completely open–topic conversations. We also find a correlation between cumulative sentiment (using sentiment and dialogue acts) and user ratings and argue that this could be investigated to estimate an error–signal for strategy computation.

1 Introduction

The structure of human dialogue is closely connected to the underlying goals of the participants to the conversation. Therefore, having a model of the speaker's goals represents the only way to have a form of control (as well as understanding) on the decision–making process of the machine in human–computer conversational interactions.
Traditional task–oriented approaches to dialogue generation[21] solve this issue by circumscribing the application of the dialogue system to a specific domain (e.g. restaurant reservations); and usually rely on carefully hand–crafted parameters, such as the state and action space for the domain. Such constrained settings allow the system to be in good control of the interaction at each turn, representing the user input through intents (the goals of the user) and their associated slots (entities mentioned in the conversation); and allow using modular architectures [20]. However, given the dependency on either domain–specific data or hand–crafted features, this approach is difficult to scale to open–domain conversations.
A different line of research emerged in recent years which replaces the modular pipeline of traditional systems with a single model (e.g. sequence-to-sequence recurrent neural networks [17, 4]) trained on a large collection of dialogues. Such models directly generate the most probable response to a user utterance. Since the most probable responses tend to be the most generic and dull ones, while the most

1st Proceedings of Alexa Prize (Alexa Prize 2017).

diverse and interesting ones are naturally much more sparse [13], such systems are hardly engaging. Due to the fact that sequence-to-sequence models learn a direct association between string sequences, there is no need for hand-crafted features or explicit representations of user intents. However, this lack of explicit representations makes them difficult to interpret and control; and hard to keep the conversation context for more than a few turns. Consequently, the model might end up in loops of dull responses [14].
Sequence-to-sequence models have the benefit of being domain-independent given the right data set. Traditional modular domain-specific task-oriented architectures have the benefit of control and interpretability of their decision–making. However, the design of open-domain and controllable conversational agents still remains an open issue. Additionally, user engagement in human-machine interaction, crucial for open-domain conversations, is a rather new research area addressed by very few [22]. Moreover, regardless of the approach, there is a lack of large and high-quality data to train domain-independent conversational systems.
In Roving Mind (RM) the advantages of both approaches are joined into a system which is at the same time modular and domain–independent, with a specific submodule to address user engagement. In particular the characteristics of our architecture (shown in Figure 1) are:

- *modularity:* in RM the behavior of each module is not affected by others. This allows firstly to interpret and control the decision–making process of each submodule and secondly to easily combine and replace rule–based approaches (currently required by the lack of appropriate training data) with data–driven ones. Moreover, the modularity of the approach allowed to add an *Engagement* submodule to our architecture with the task of keeping the user engaged in the interaction. RM's architecture can therefore be described as a balance between human–expert design (for the global architecture) and data–driven approaches (locally).
- *domain independence*: we represent both user and machine utterances through domain-independent *functional unit* (FU) structures, consisting of a list of *entities* (slots in task-based approaches) and a *dialogue act* (DA) structure representing the goal of that FU (intents in task-oriented approaches). Our system also supports queries to various domain-independent knowledge bases (KBs), such as news, useful for user engagement. In addition RM has been designed to support both single- and multi-turn strategies (e.g. news summaries or predefined subdialogues).

To the best of our knowledge RM represents the first attempt to build a modular, domain-independent dialogue system architecture, with an explicit representation of user goals and the entities mentioned in the conversation and a fully automatic module for user engagement.
After detailing the system architecture in section 2, in section 3 we analyze the results of the Alexa Prize semifinals, during which RM was tested and rated by a large group of Amazon Alexa customers. While our initial setting during the competition was completely open-topic, we report the results of a set of experiments performed in order to optimize user ratings using features aimed at making the conversation more driven towards the topics where we had the highest coverage, without changing our architecture. We have observed a significant correlation between user ratings and *cumulative sentiment* – a combination of sentiment and DAs revealing sentiment of the user towards a topic – which could be used in the future as a potential error–signal for user engagement. Finally in section 4 we draw on the conclusions of our work.

2 System design and architecture

In this section we will go through a coarse-grained description of RM's scalable infrastructure pipeline (section 2.1) and its KBs (section 2.2). Subsequently, we describe the design and implementation of the Spoken Language Understanding (SLU) pipeline in section 2.3 and the Dialogue Manager (DM) and Natural Language Generation (NLG) modules in section 2.4.

2.1 System pipeline

The input of the system pipeline is the data structure coming from the Alexa Skill Kit Automatic Speech Recognition (ASR) module. The first element of the pipeline is the client application, whose main goal is to address predefined intents of the Alexa Skill Kit and redirect everything else to the server, which will generate the response. The client application was implemented using an AWS Lambda function, as in common practice in Alexa Skills implementation.
The server exposes a RESTful API to communicate with the client. Its first task, once the request is

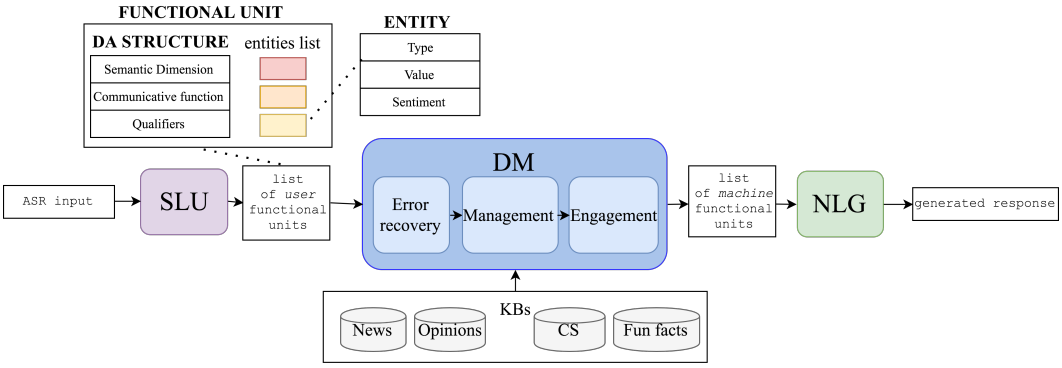

Figure 1: In our architecture the Dialogue Manager (DM) takes as input a list of *user* functional units (FUs) (i.e. consisting of an associated dialog act (DA) structure and a list of entities) created by the Spoken Language Understanding (SLU) module from the Automatic Speech Recognition (ASR) output. This list is processed progressively by each of the three DM submodules to create a list of *machine* FUs (also retrieving information from a set of knowledge bases). This list is then passed to the Natural Language Generation (NLG) module, where the corresponding template for each machine FU is generated and joined together to produce the final response.

received, is to interact with a database to retrieve the dialogue history collected during the interaction. After this information is retrieved, the server runs through the SLU pipeline to create the list of user FUs and then passes the results to the DM module, which produces the output machine Dialogue State. During the DM phase, AWS RDS is used to manage the KBs. These KBs are constantly updated through independent scripts running on separate EC2 instances. The last step of the pipeline is the NLG module, which transforms the generated list of machine FUs into a string output. This output is finally returned to the client application, which sends the text to the Alexa Text-To-Speech (TTS) provided by the Alexa Skill Kit.
The server API was implemented using an AWS Elastic Beanstalk, which eases the scalability of the system and makes it more robust and reliable, while the database containing users data was implemented with an AWS DynamoDB, which allows high read–write throughput.

2.2 Knowledge Bases

Roving Mind has a rich set of content sources to keep the user engaged and to maximize the coverage of the topics provided by the user. In its final version, the system features News, external Opinions, Fun Facts and a knowledge graph to perform Commonsense reasoning. Factual question answering (Q&A) is performed querying the EVI answer engine.

News The News database is based on two main scripts that run continuously in background, updating the KB every day. The first script downloads the news from the Washington Post API and stores them in an AWS Aurora database. In this phase news are clustered together, according to the 'section' parameter provided by the API. An affinity propagation[11] clustering technique is used to group news efficiently and enable a fast search for related news on the same domain. The second script processes the downloaded news articles for both generating article summaries and linking news to a specific keyphrase. To be able to link news articles to keyphrases, sentences in an article are represented as lists of triplets in the form <subject,predicate,object>. The approach was adapted from the ClausIE framework [9]; and allows to classify each sentence as relevant or not to a keyphrase.

Opinions Opinions are extracted from news articles using Conditional Random Fields (CRF) models trained on the Parc dataset [15]. This model extracts quote spans and attribute them to their authors. The obtained quote spans are processed for keyphrase extraction and triplet generation. Each quote is then classified as relevant or not to the keyphrase according to the keyphrase's position in the triplet. Finally, in the KB, the quote is represented with the disambiguated speaker of the quote, the span and the main keyphrase.

Commonsense For our third KB, representing commonsense, we use the data from ConceptNet 5 [18] and integrate it with data regarding entities connected to the relation "occupation" from Wikidata.

With the collected data we created a graph. On the resulting graph we defined a set of queries that are performed online during the execution of commonsense strategies (see section 2.4).

Fun Facts Fun facts were collected online[1]. Similarly to news, we extract keyphrases and generate lists of triplets in order to trigger fun facts related to a given keyphrase mentioned by user.

2.3 Spoken Language Understanding

Spoken Language Understanding (SLU) for task-based systems is traditionally approached as identification of the intent (the purpose behind the utterance) and detection of its arguments (slot-filling) from a user utterance [19]. In a task and domain-specific SLU the set of intents and slots is usually predefined; in a domain-independent SLU, on the other hand, it is not the case.
Since the range of user intents in an open-domain dialogue is virtually unbound, in RM intents rely on domain-independent dialog acts (DA) (e.g. questions, requests), and are further specified with an additional set of qualifiers. Intents cover various aspects of a conversation, as they can represent information exchanges (e.g. questions and points of view expressed by the user), social obligations (e.g. thanking, salutations) and action discussions (e.g. requests, orders). In addition, a user utterance can contain one or more intents: e.g. greeting and an action request; thus, an utterance needs to be segmented with respect to these intents. The recently accepted international ISO standard for DA annotation – DiaML (ISO 24617-2) [5] – is a good choice to address these necessities. An utterance is segmented into functional units (FU) and each unit is attributed a DA that consists of semantic dimension (e.g. Social Obligation) and communicative function (e.g. Thanking). Thus, the specification naturally lends itself to the design of open-domain systems. In RM, the ISO specification was adapted, following in the work of [8], with emphasis on the communicative aspect rather than the semantic dimension.
Similarly to intents, in a domain-independent non-task-based setting the range of possible slots is unlimited. In RM, we replace the notion of slot, predefined in task-based systems, with the one of entity. An *entity* is defined as any user-provided content that can function as a topic of a conversation – Named Entities, keyphrases, etc. – or that triggers specific dialogue functionality – News, Opinions etc. Entities are not predefined, but classified into types according to their association to our KBs.
Hence, in RM each FU is represented by an intent, that is a DA structure composed by a DA (identified by semantic dimension and communicative function) and its qualifiers, and a set of entities extracted from the span of that FU (see Table 1).
The rest of the section describes the system pipeline for the segmentation into functional units and identification of user intents, and entity extraction.

Pre-processing: The SLU pipeline starts with the pre-processing of user input. First, the ASR output is case-restored to improve the detection of Named Entities. Then, the input is split into sentences using the state-of-the-art sentence-boundary detection algorithm of [16], without prosodic information (not available). Each detected 'sentence' is used to extract information for a FU structure.

Intent Identification: The first step in intent identification is DA tagging. For DA tagging we consider three semantic dimensions from DiaML [5]: (1) Social Obligation dimension that addresses basic social conventions such as greetings, (2) Feedback dimension that addresses user's feedback regarding the previous machine statement, and (3) Task dimension that addresses user's actions such as requests for information or action.
Given that each of the semantic dimensions consists of specific set of communicative functions, the DA tagger is implemented in two separated steps. We first identify the dimension of the DA, and then use dimension specific classifiers to identify the communicative function. For the Social Obligation dimension, the communicative functions are assigned using a lexicon-based system. Functions from this dimension capture salutations, apologies and thanking, following the ISO standard definition. For the Feedback dimension, on the other hand, there is only one communicative function which is simply called 'Feedback'. The Task dimension is more complex and, in RM, consists of 16 communicative functions, which correspond to the "General Function" tags of the ISO standard scheme, where some of the lower level tags are not considered. These tags are arranged in a tree structure: at the first level there is a separation between Action discussion, which captures specific dialogue-related actions by the user (e.g. Stop, Repeat, Start Quiz), and information-transfer, which captures an information exchange (statement or question) between the user and the system. Lower levels provide more details about the interaction, specifying for example whether it is an Information providing or seeking (for

[1] `https://www.thefactsite.com/`

Information transfers) or whether the action performed by the user is Directive or Commissive (for Action Discussions).
DAs are further analyzed to determine sentiment polarity, factual information type and subjective information type (i.e. qualifiers). The sentiment polarity qualifier records the attitude of a user towards the entities (or topics) present in FU, sentiment analysis is performed using a lexicon-based analyser following [2]. The factual information type qualifier is similar to the Expected Answer Type in Q&A and records if the provided/requested information is about a specific time, place, reason, entity, etc. The subjective information type qualifier, on the other hand, categorizes non-factual information such as opinions or personal information (e.g. "My favorite sport is football"). The qualifier assignment is handled with lexicon and rule-based systems.
Finally, a lexicon based functionality classifier determines whether the user dialogue act contains a request for a specific functionality of the system: such functionalities are divided in (1) dialogue control functionality: built-in intents such as Stop, Repeat and Continue; and (2) Content Request functionality: requests for one of the featured contents of the system such as News or Fun Facts.
DA taggers were trained using multiple corpora – Switchboard[12], AMI [6], the HCRC Maptask corpus[3], the BT Oasis Data and the VerbMobil2 [1]. Following the approach of [10], the DA annotation of these corpora was mapped to the modified tag hierarchy of RM. We use a combination of CRF and Support Vector Machine classifiers for tagging; as it achieved the highest performance on the withheld 10% of data.

Entity Extraction and Linking: RM extracts Named Entities and keyphrases which are later linked to nodes in the KBs. Keyphrase extraction is performed via a simple Noun Phrase Chunker. After an initial filtering step to remove phrases containing 'taboo' words (specified by a stop-word list), extracted chunks are linked to KB nodes using the approach of [7]. Commonsense nodes extraction is performed via a Breadth First Search on the dependency parsing tree for the user utterance, starting from the root, down to the leaves. All paths are explored, and sibling nodes sharing common properties become candidate nodes. When all candidates are collected, each of them is verified against the label present in our commonsense KB and, if a match is found, it is stored in the FU structure. The list of *user* FUs generated by the SLU is then passed to the DM.

2.4 Dialogue Manager and Natural Language Generation

The Dialogue Manager of Roving Mind is modular (see Figure 1), and consists of three main components, each addressing a specific dialogue function: the *Error Recovery*, the *Management* and the *Engagement* modules. These components work in a sequence, each taking as input the output of all previous modules, i.e. the list of user FUs [5], the list of generated machine FUs so far, and the entire history of the conversation in terms of strategies, entities and FUs selected in previous turns. Each DM module can generate zero, one or more new FUs and add them to the list of machine FUs which will then be passed to the NLG to generate the final turn. The only difference between user and machine FUs is that each machine FU is associated with a specific strategy and the relevant content, retrieved from the KBs to fill the NLG template for that strategy. In RM, the list of current user FUs, the list of current machine FUs and the previous dialogue history constitute a dialogue state.
Internally, all the three submodules are based on similar components, most of which are currently rule–based due to the lack of training data, but are easily replaceable by data–driven modules thanks to the modularity of the architecture. There are two main phases for each module: *machine FU creation* and *content retrieval*. In the first phase, taking as input the list of current user FUs, the list of current machine FUs and the previous dialogue history, the system can (but it's not mandatory) create a new machine FU with its related machine DA structure, entities and strategy. Starting from this partially formed FU, in the second step the system tries to instantiate the selected strategy by retrieving the content required by the strategy through queries on the available KBs. If the content retrieval phase fails the control is returned to the *machine FU creation* step which can create another machine FU selecting a different combination of strategy/entities/DA structure.
Overall we designed more than 60 dialogue strategies (mapped to DAs), each one selectable by one or more modules, grouped among error–recovery, dialogue–management, Q&A and engagement strategies. As mentioned, some of our strategies rely on queries on our KBs (e.g. the ones providing relevant news summaries given an entity of type Keyphrase). Probably the most interesting set of strategies were the ones relying on commonsense, selectable in case an entity of type Commonsense node was found. Firstly we created a list of categories of nodes relevant for open–domain conversation (e.g. human activity, entertainment event) from our commonsense KB. Each category was identified

by a set of commonsense relations that nodes belonging to the category had to fulfill. Then we created a dataset of conversational natural language templates each one based on one of the selected categories. Each template was tagged with an utterance type (question or statement) and a DA structure (e.g. its communicative function etc.) as described in 2.3. Given as input from the DM a list of commonsense node entities, the DA structure and the utterance type, our commonsense strategies first select the list of templates matching that DA structure and utterance type. Afterwards, if any templates were found, the strategy verifies that the input nodes belong to the category appearing in the templates and finally, if additional commonsense queries are required to fill the template, it executes them online. At last, if any templates were generated, the most relevant template is chosen (i.e. the one sharing the higher number of commonsense entities with the original), otherwise the strategy fails and the control is returned to the DM.
Given the basic two steps previously described, each DM submodule works in a slightly different way according to its main goal.

Error recovery module The Error Module is the first module of the pipeline to be executed. The traditional function of the Error Recovery module is to trigger a strategy for user FUs with low ASR confidence. We extended the goal of this module to a more generic analysis of user FUs that are not "well–formed" according to "explicit" content (using a dictionary) and personal information (using Regex). This decision is due to the fact that the system handles these cases in a similar manner, filtering them from the FUs to store. The strategies this module can select from are traditional error–recovery strategies (e.g. asking the user to repeat in case of low ASR).

Management module Once the input has been filtered, the output is passed to the Management module, whose main goal is to address the potential requests (including questions) expressed by the user in the last turn (backward-looking). Together with the Error module the Management can be used independently from the last module as a Q&A system or a task–based system. The second goal of this module is to update the active entities in the system, by adding any entity the user mentioned in the input utterance to the entity pool of the current dialogue state. Moreover, if the user utterance has a negative sentiment towards entities introduced by the system during the previous turn, they are discarded. In addition, this module manages the continuation of multi–turn strategies (such as subdialogues) started by the system in previous turns.
In this module, during the *machine FU creation* step each user FU is mapped to a machine FU through a rule-based approach. Afterwards, in the *content retrieval* step, each machine FU is associated to a set of strategies (applied in a sequence) designed to address that specific request (e.g. an entity of type question is sent to the Q&A strategies). If the system detects a user FU that cannot be handled by the system, the *content retrieval* step will generate a machine FU with a strategy informing the user that the required functionality couldn't be handled. If every strategy fails, the system creates an empty machine FU that will be used by the Engagement module to determine the next action and discarded later on. The same approach is applied if the user FU does not contain any requests/questions. Regarding the Q&A strategies, questions detected as factual are redirected to Evi, while for personal questions we use the previously described commonsense strategies.

Engagement Module After all requests have been satisfied the output is passed to the Engagement module, responsible for keeping the user engaged in the conversation (forward-looking). The idea of having engagement as a module in a dialogue system design was introduced in [22], where engagement is defined as the user's interest in continuing the conversation. The authors shows how having a module which addresses directly user engagement leads to a better user experience. The difference with [22] is that our engagement module is fully automatic and takes into account the machine FUs created by previous modules for the current turn.
The Engagement Module takes as input the utterances generated by previous modules with the updated history of active entities and the history of the conversation. In the *machine FU creation* step, the module firstly verifies whether machine FUs created by the previous modules require additional engaging content. If that is the case, entities from the history are sorted according to a series of criteria, which include the types of the available entities, whether they are user-introduced or machine-introduced and their age in the system. Once entities are ranked and the entities to be used is decided, the list of engagement strategies is filtered by selecting the ones compatible with the type of entities. Then the strategies are ranked in a rule-based submodule according to three main features: the current turn index of the conversation (e.g. some strategies work better if they are run in an early/late stage of the conversation), the amount of times the strategy has been used in the previous turns and the ratio between questions and statements in the dialogue history (in order to

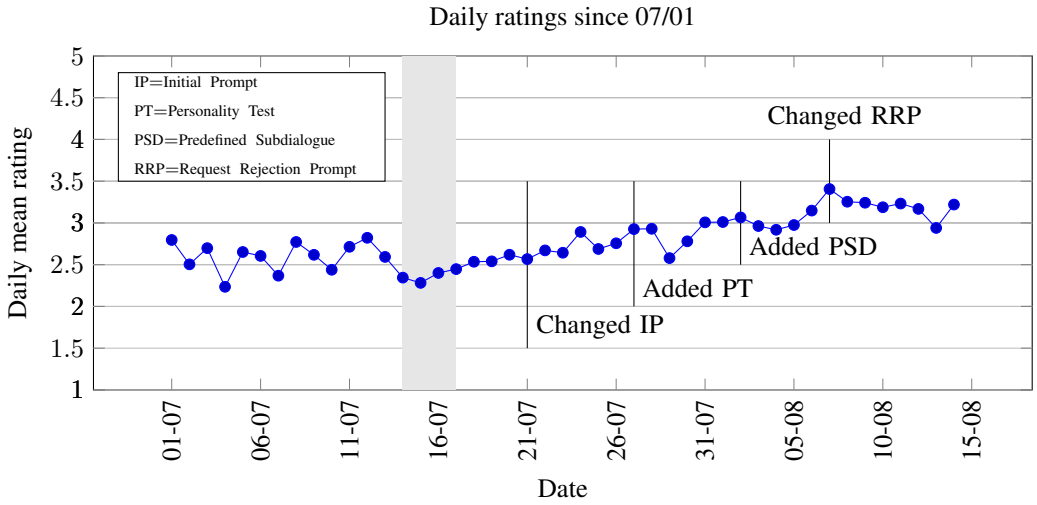

Figure 2: The figure shows the effect of the positive effect of our experiments on our daily average ratings. The daily volume of rating was constant with small daily variations, except for the period between 14th and 17th of July (in gray) were we had an increment of about 289% of rated interactions compared to the daily conversations average.

keep a balance between them when interacting with the user). Finally, in the *content retrieval* step, the selected strategies are executed according to their ranking until one of them succeeds. If all of them fail, a different set of entities is selected and the process is repeated all over again until either a strategy succeeds or the maximum number of attempts is reached, in which case a entity-less strategy is selected.

NLG The last step of the pipeline is the NLG module, which receives as input a list of machine functional units, each one with a corresponding strategy and the related content retrieved from the Contents KBs. For each of these FUs the NLG generates an output utterance filling the template associated to that strategy with the related content. All the created utterances are then concatenated in the final response and passed to the TTS.

3 Evaluation

During the Alexa Prize semifinals, Roving Mind was tested for 6 consecutive weeks by Amazon Alexa users, which at the end of the interaction were asked to rate how likely they were to interact with that socialbot again on a 1–5 scale.
The initial RM dialogue system (active till 21st of July) was completely open – there was no bias towards any domain or topic. In the following weeks, a set of new functionalities was added such that each new functionality was evaluated for a short period of time and compared to the previous system ratings. The dates and results of these changes are visible in Figure 3. The details of the experiments are discussed in Section 3.1.
The new functionalities evaluated during semifinals are the following: (1) driving conversation towards a specific set of domains/topics via initial prompt, (2) addition of "entertaining" features such as quizzes and tests, (3) system-driven subdialogues, and (4) change to the request rejection prompts.

3.1 Experiments

Open Prompt vs Directed Prompt Our first experiment was changing the opening prompt from a very generic "What do you want to talk about" to "We can talk about topics like sports, politics or celebrities. What would you like to chat about ?". This was done after noticing a large number of conversations in which users appeared confused about the features of the system and were expecting a list of possible topics, rather than coming up with their own. Also, the change of opening prompt gave us the possibility of driving the user towards topics where Roving Mind has a high coverage across different strategies. To evaluate the results of the change we drew a comparison with the

System		Time Window		Mean Rating
RM	(baseline)	Jul 15 - Jul 20	(6)	2.44
RM+IP	(+initial prompt)	Jul 21 - Jul 26	(6)	2.75*
RM+IP+PT	(+personality tests)	Jul 27 - Aug 01	(6)	2.89*
RM+IP+PT+PSD	(+predefined subdialogue)	Aug 02 - Aug 07	(6)	3.07*
RM+IP+PT+PSD+RP	(+rejection prompt)	Aug 08 - Aug 15	(8)	3.17

Table 1: Changes to the Roving Mind system with the time window for data collection (number of days in parentheses) and mean user rating received for the rated conversations within that window. The ratings for each version of the system are compared to the previous one. The statistically significant changes in rating are marked with * ($p < 0.01$).

previous week ratings, as visible in Table 1. Results are statistically significant and show that average rating has improved with the simple addition of a driving opening prompt.

Personality Tests and Quizzes Personality tests and Quizzes have been added to evaluate the effect of simple user–entertaining features on user ratings without any modification of our architecture. The five personality tests and the only quiz proposed share the same simple structure, at the beginning the quiz or test is proposed to the user, if the user accepts the offer, the system will go through a set of predefined questions asking the user to select one among multiple choices. The quiz is interrupted by any question and request made by the user. After three or four questions the system provides the result of the test. In the window considered, the personality test is proposed according to the entities introduced by the user or a random one when the user does not introduce any entity. We decided to compare the ratings of conversations during a week with personality tests and during the previous week, when tests were not yet part of the system. Results, as visible in Table 1, show a high statistically significant difference between the rating before and after the introduction of quizzes.

Predefined subdialogues The results of the personality tests seemed to suggest a positive effect of predefined multi–turn structures on user ratings. For this reason we created a set of predefined subdialogues where the user is guided through a sequence of coherent prescripted responses and decided to verify their effect on user ratings. The structure of subdialogues is similar to the one of personality tests and quizzes. The main difference is the fact that the system adds a reaction to the user statement, supports a set of predefined question related to each machine turn and according to the sentiment of the user response can choose a different question for the next turn. This setting allows to have a tight control on the user interaction during the subdialogue. We defined a small set of subdialogues that do not require any entity introduced by the user that start at the beginning of the interaction and another set of subdialogues triggered when a set of specific entities was mentioned in the conversation. The results show a boost in ratings as shown in Table 1. However, the system achieves the best performances when the user experiences both subdialogues and quizzes together, as visible in Table 2.

Changed Request Rejection Prompt Given that in previous experiments guiding the user through the conversation had a positive effect on user ratings, we decided to change the prompt used by the machine in case the requested feature was not supported. Instead of simply saying that the system was not able to satisfy the request, the changed prompt provided alternatives on our system's capabilities. This change had a positive effect on user ratings as visible in Table 1.

User Interaction and Ratings: In the course of evaluation users were providing *subjective* ratings, which are expensive to obtain while developing conversational systems. An objective metric that could be used as an online error signal for user engagement towards a topic or a DM strategy is a sentiment score, already computed in RM. Cumulative average sentiment is computed as the average of user FU sentiments combined with the sentiment expressed by DAs revealing the user's sentiment towards a topic (e.g. yes/no answers to topic proposals), sentiment analysis is performed as described in section 2.3. Since engagement is also reflected in the conversation length, it is another evaluated objective metric. Analysis of user ratings with respect to conversation length and cummulative sentiment over the duration of conversation reveals statistically significant differences between low-ranked (rating 1-2) and high-ranked (3-5) conversations using t-test and $p < 0.01$. Low ranked conversation had an average conversation length of 9, while high ranked ones of 13 turns. Moreover, sentiment score that considers only DAs yields statistically significant differences as well.

Dialogue Subset	Mean Rating
no personality test or predefined subdialogue	2.66
personality test **or** predefined subdialogue	3.08*
personality test **and** predefined subdialogue	3.71*

Table 2: The effect of the activation of predefined subdialogues and personality test on user ratings, between July, 21st and August, 10th. Each strategy activation setting has been compared to the row above, and ratings with statistically significant differences are marked with * ($p < 0.01$).

4 Conclusions

We presented a modular open–domain dialogue system which reacts to user engagement. Our experiments suggest that a fully open-domain, non task-based dialogue system makes users confused and tends to produce lower ratings. This emerged by the progressive success of all our experiments (prompt change, personality tests, prescripted dialogues and prompt for unsupported request change). Another key aspect of the interaction with the user that came out during the semifinals is that users do not necessarily expect a regular, conversation-based interaction with social bots: many users are pleased with non canonical discussions such as personality tests, quizzes and word games, especially if they are coherently inserted in a conversation.
These results could be due to the fact that usually Alexa skills are based on strict, predefined interactions, and users expect the same kind of interaction from the Alexa Prize socialbots. In addition, our analysis shows that users have an average of 1.07 conversations, with a variance of 0.14, which could explain how strategies like personality tests and scripted dialogues kept giving good results throughout the competition, despite the fact that these strategies would probably annoy the users if proposed more than once. Another interesting finding was the positive correlation between both average length of conversations and average user's cumulative sentiment and user ratings. This suggests that users tend to give higher ratings to longer conversations and conversations in which they discuss entities towards which they have a positive attitude.
Regarding the future work, one improvement of our architecture would be the replacement of rule-based modules with data–driven ones using data collected during the competition, exploiting the modularity of our system. Useful features to train these modules could include successful patterns of strategies found in high-rated conversations, as well as information about the sentiment which emerged from the analysis of the user logs. We could also exploit our findings regarding cumulative sentiment to develop an error signal to integrate in our architecture. Another important improvement would be to enrich the commonsense knowledge base with other Wikidata information besides the "occupation" relation (e.g. movies, videogames, books), while at the same time refining the way entities and relations introduced by the machine are selected, for example ranking entities according to how much they compare in our news KB in order to select entities users are familiar with. Finally, given the success of our experiment on predefined subdialogues, we plan to design an automated system to generate subdialogues, exploiting the ranking of entities across our knowledge sources (for example the news of the day).

Acknowledgements

We would like to thank Evgeny Stepanov, Arindam Ghosh and Ali Orkan Bayer for their invaluable suggestions throughout the competition.

References

[1] Jan Alexandersson, Bianka Buschbeck-Wolf, Tsutomu Fujinami, Michael Kipp, Stephan Koch, Elisabeth Maier, Norbert Reithinger, Birte Schmitz, and Melanie Siegel. *Dialogue acts in Verbmobil 2*. DFKI Saarbrücken, 1998.

[2] Kennedy Alistair and Inkpen Diana. Sentiment classification of movie and product reviews using contextual valence shifters. *Proceedings of FINEXIN*, 2005.

[3] Anne H Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. The hcrc map task corpus. *Language and speech*, 34(4):351–366, 1991.

[4] Ali Orkan Bayer, Evgeny A. Stepanov, and Giuseppe Riccardi. Towards end-to-end spoken dialogue systems with turn embeddings. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Stockholm, Sweden, August 2017. ISCA.

[5] Harry Bunt, Alex C Fang, Xiaoyue Liu, Jing Cao, and Volha Petukhova. Issues in the addition of iso standard annotations to the switchboard corpus. In *Workshop on Interoperable Semantic Annotation*, page 67, 2013.

[6] Jean Carletta. Announcing the ami meeting corpus. *The ELRA Newsletter 11(1), January-March, p. 3-5.*, 2006.

[7] Andrew Chisholm and Ben Hachey. Entity disambiguation with web links. *Transactions of the Association for Computational Linguistics*, 3:145–156, 2015.

[8] Shammur Absar Chowdhury, Evgeny A Stepanov, and Giuseppe Riccardi. Transfer of corpus-specific dialogue act annotation to iso standard: Is it worth it? In *LREC*, 2016.

[9] Luciano Del Corro and Rainer Gemulla. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366. ACM, 2013.

[10] Cao J. Bunt H. Fang, A. and X. Liu. The annotation of the switchboard corpus with the new iso standard for dialogue act analysis. *Proceedings of ISA-8, Pisa, pp. 13-18.*, 2012.

[11] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[12] John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 517–520. IEEE, 1992.

[13] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pages 110–119, 2016.

[14] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

[15] Silvia Pareti. A database of attribution relations. In *LREC*, pages 3213–3217, 2012.

[16] Jonathon Read, Rebecca Dridan, Stephan Oepen, and Lars Jørgen Solberg. Sentence boundary detection: A long solved problem? *COLING (Posters)*, 12:985–994, 2012.

[17] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[18] Robert Speer and Catherine Havasi. Representing general relational knowledge in conceptnet 5. In *LREC*, pages 3679–3686, 2012.

[19] Gokhan Tur and Renato De Mori. *Spoken language understanding: Systems for extracting semantic information from speech, Chapter 3*. John Wiley & Sons, 2011.

[20] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2016.

[21] Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[22] Zhou Yu, Leah Nicolich-Henkin, Alan W Black, and Alexander I Rudnicky. A wizard-of-oz study on a non-task-oriented dialog systems that reacts to user engagement. In *SIGDIAL Conference*, pages 55–63, 2016.