Robot Data-Driven Imitation Learning of Human Social Behavior with Time-Persistent Interaction Context

MALCOLM DOERING

SEPTEMBER 2019

Robot Data-Driven Imitation Learning of Human Social Behavior with Time-Persistent Interaction Context

A dissertation submitted to
THE GRADUATE SCHOOL OF ENGINEERING SCIENCE
OSAKA UNIVERSITY
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY IN ENGINEERING

BY

MALCOLM DOERING

SEPTEMBER 2019

Abstract

Imitation learning as a method for designing interactive behaviors for social robots has been increasingly explored in recent years [78, 79, 80, 81]. This approach to creating social robot behaviors has several advantages, such as requiring little input from human designers or data-annotators and learning action policies that are robust to noise and natural variations of speech. Furthermore, the early work, which originally proposed this approach, successfully demonstrated systems that learned to replicate the behaviors of a human shopkeeper in a camera shop scenario [78, 79, 80, 81]. However, those systems decided the robot's actions based on only the current, immediately observable interaction state, without reference to events in the interaction history. This limited the behaviors that the robot could perform.

To surpass the earlier limitations, this thesis presents new methods for imitation learning of human social behavior with *time-persistent interaction context*. That is, methods for automatically deciding the robot's action based information about the interaction history are presented. Specifically, three sub-problems of robot imitation learning of human social behavior are explored, and techniques for modeling time-persistent interaction context that solve those problems are proposed.

In the **first study** the earlier data-driven approach is extended via a recurrent neural network architecture (RNN) to enable the shopkeeper robot to generate actions that depend on memories of events in the interaction history. First, an analysis of *memory-setting* and *memory-dependent* actions in a human-human camera shop dataset is presented. Then, a large interaction dataset is simulated to evaluate the system. An RNN architecture with Gated Recurrent Units (GRU) that automatically learns from unlabeled data to generate appropriate memory-dependent actions is proposed. Finally, the proposed architecture is analyzed to understand the GRU's memory representation.

In the **second study** the earlier approach is extended with a time-persistent customer behavior model, online learning, and a curiosity mechanism to enable a shopkeeper robot to adapt to individual customer differences and generate more varied, interesting behaviors. In an evaluation, the proposed system was able to learn to better model customer behaviors through (simulated) interactions with customers. In a user study, participants found a curious robot to be significantly more humanlike with respect to repetitiveness and diversity of behavior, more interesting, and better overall in comparison to a non-curious robot.

In the **third study** a novel robot behavior learning system that models the time-persistent topic of conversation for the purpose of dealing with ambiguous customer questions is presented. A new dataset of human-human interactions in a travel agent scenario was collected to evaluate the system. A novel topic clustering algorithm based on action co-occurrence frequencies and a method for topic estimation during runtime are presented. The proposed technique outperformed several baseline techniques in qualitative and quantitative evaluations. It responded more accurately to ambiguous questions and participants found it was easier to understand, provided more information, and required less effort to interact with.

Table of Contents

Chapter 1

Introduction

Nowadays, social robot platforms are becoming more available in the world, but the question of how to design behaviors and interaction logic for these robots to introduce them into new domains remains a challenge.

There are at least three broad categories of approaches to designing social robot interaction behaviors - cognitive architectures, such as ACT-R/E [126] and SOAR [67], that use first principles to respond to stimuli from the environment and take actions based on modeling and understanding the world; hand coding behaviors using if-then style rules or editors like Choreograph [97] and Interaction Composer [31, 32], such that all the interaction content is scripted by a designer; and data-driven approaches in which the robot imitates the outward behaviors observed in human-human, or teleoperation, example interactions, without requiring any model of the world or input from a human designer. The current work focuses on an approach in the last category.

1.1 Data-driven Imitation Learning of Human Social Behavior

The details of the data-driven imitation learning approach to human-robot interaction and its advantages have been thoroughly described in related works [78, 79, 80, 81]. Here I present an overview.

The main idea of the data-driven approach to HRI is to 1) collect a dataset of examples of natural human-human interaction in a target domain (e.g. a camera shop or travel agency). Then, 2) apply data abstraction and discretization techniques, including clustering and models of HRI formations, to the raw data to make the learning problem more tractable. Next, 3) apply machine learning algorithms, such as clustering and classification algorithms, to the dataset to automatically learn social robot behaviors and interaction logic without manual-annotation of the data. Finally, 4) replace the target human (e.g. the shopkeeper or travel agent), with a social robot that uses the learned behaviors and interaction logic to interact with human customers.

The primary motivation for a data-driven approach to HRI is to quickly design so-

cial robot behaviors for new domains without expensive, time-consuming hand-coding or manual data annotation. Furthermore, it is not always possible for an interaction designer / programmer to anticipate the myriad ways that a human may behave in interaction with a social robot, but by learning behaviors from datasets that contain many examples of natural interaction behaviors, the robot can learn to respond to various unanticipated customer behaviors. Another advantage of the data-driven approach is that by training the robot on interaction data collected using the same sensor array as the robot will use to sense the environment, it can learn interaction logic that is robust to sensor noise.

The earlier work has successfully demonstrated the ability of a camera shopkeeper robot, trained using the data-driven imitation learning approach, to interact naturally with human customers in a simplified camera shop scenario [78, 79, 80, 81]. While this is an incredible achievement, this work has some limitations.

1.2 Limitations of the Earlier Approaches

The systems in earlier work decided the robot's actions based on only the current, immediately observable interaction state, without reference to events in the interaction history. This limited the behaviors that the robot could perform.

In the earlier work [78, 79, 80, 81], the robot's actions were decided based on the output of a classifier that takes the current *interaction state* (customer and shopkeeper's locations, motion target, motion origin, utterances, and spatial formation) as input. This setup limits the robot's interaction logic because contextual information from before the current turn interaction was not preserved in the interaction state. The absence of information about the interaction history limits the types of behaviors that the robot can learn to imitate.

For example, consider the following types of behaviors that robot systems in the earlier work cannot replicate:

- **Actions that depend on events in the interaction history**. For example, in order to make appropriate recommendations to customers, a robot system must learn to remember when the customer states their preferences.

- **Adaptation to individual customer's differences**, such as interaction style. The quality of human-robot interaction improves when the robot is able to adapt to individual customer differences, for example being proactive with an engaged customer vs. leaving alone an unengaged customer who merely wishes to browse. To adapt in this way, the robot system must remember how the customer behaves throughout the interaction, i.e. learn a time-persistent model of the customer's behavior.

- **Responding to ambiguous customer questions**. To respond to ambiguous customer questions the robot system must comprehend topic the of conversation,

which is not always immediately observable in the current interaction state (e.g. in the content of the customer's most recent utterance). Thus, the robot must calculate a time-persistent estimate of the topic of conversation.

1.3 Time-Persistent Interaction Context

Context is an important concept in a variety of fields including linguistics, human-computer interaction, and anthropology. Duranti and Goodwin define context as:

> ... a frame [35] that surrounds the event being examined and provides resources for its appropriate interpretation ... The notion of context thus involves a fundamental juxtaposition of two entities: (1) a focal event; and (2) a field of action within which that event is embedded. [25]

This is a *general* definition of context. In this thesis we are concerned with a more specific type of context, which is most closely related to *context* as used in the study of dialog systems and discourse analysis.

Context's function in dialog is to help the hearer interpret the meaning of the speaker and for the speaker to decide his next action. Previous works have introduced various categorizations of context in dialog and discourse. For example, Bunt suggests the categories of *semantic context*, *processing status*, *linguistic context and dialog memory*, *physical/perceptual context*, and *social context* for spoken dialog management systems [12]. Similarly, Song presents the categories of *linguistic context*, *situational context*, and *cultural context* [118]. Furthermore, she states that context's function in discourse includes eliminating ambiguity, indicating referents, and detecting conversational implicature.

We define *time-persistent interaction context* in human-human and human-robot interaction roughly as the context of events that occur within the interaction and of the features of the interaction over its entire duration (in contrast to only the current moment). Moreover, these events are important for the interaction participants to interpret each other's actions and decide their own next actions. Thus, time-persistent interaction context is closely related to *linguistic context and dialog memory* above.

The main reason for the three limitations in the earlier approaches to data-driven imitation learning is that the robot's action is decided based only on the immediately observable features of the current interaction state (i.e. current interaction *context*). However, if the robot's action is decided based on the *time-persistent interaction context* (in addition to the immediately observable features of the current interaction state), then the limitations described above can be surpassed.

Therefore, this thesis presents new methods for imitation learning of human social behavior with *time-persistent interaction context*. Specifically, three sub-problems of robot imitation learning of human social behavior (corresponding to the three limitations stated above) are explored, and techniques for modeling time-persistent interaction context that solve these problems are proposed. The components of time-persistent

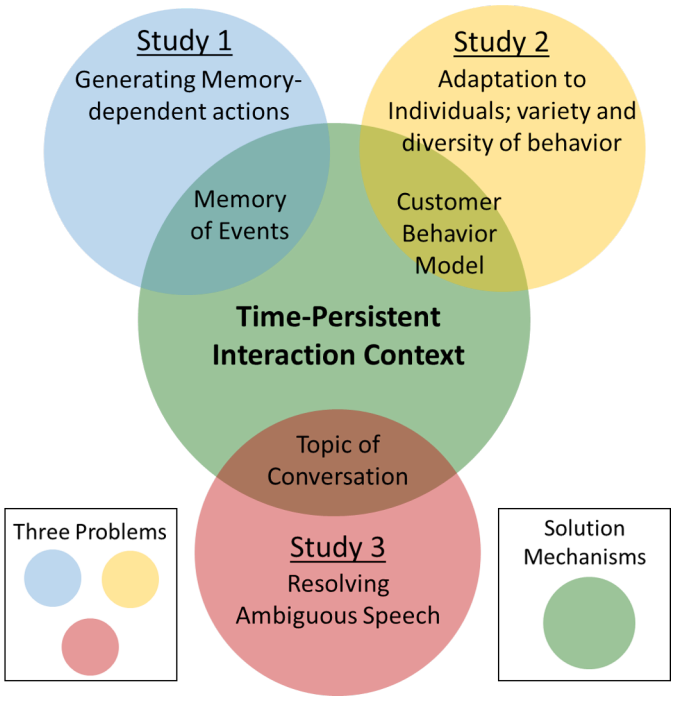


Figure 1.1: The camera shop setup contained three camera displays. Sensors on the ceiling were used for tracking human position, and smartphones carried by the participants were used to capture speech (Liu et al., 2017).

interaction context that are relevant for mitigating the above limitations are: *interaction history*, a *model of customer behavior*, and *topic of conversation*. The three studies presented demonstrate how modeling these aspects of time-persistent interaction context in the data-driven imitation learning framework can extend a social robot's abilities. A conceptual diagram of these three studies is shown in Fig. 1.1.

1.4 Organization of Sections

The remainder of this thesis is structured as follows:

Chapter 2 presents the **first study**, in which the earlier data-driven approach is extended via a recurrent neural network architecture (RNN) to enable the shopkeeper robot to generate actions that depend on memories of events in the interaction history. First, an analysis of *memory-setting* and *memory-dependent* actions in a human-human camera shop dataset is presented. Then, a large interaction dataset is simulated to evaluate the system. An RNN architecture with Gated Recurrent Units (GRU) that automatically learns from unlabeled data to generate appropriate memory-dependent actions is proposed. Finally, the proposed architecture is analyzed to understand the GRU's memory representation.

Chapter 3 presents the **second study**, in which the earlier approach is extended with

a time-persistent customer behavior model, online learning, and a curiosity mechanism to enable a shopkeeper robot to adapt to individual customer differences and generate more varied, interesting behaviors. In an evaluation, the proposed system was able to learn to better model customer behaviors through (simulated) interactions with customers. In a user study, participants found a curious robot to be significantly more humanlike with respect to repetitiveness and diversity of behavior, more interesting, and better overall in comparison to a non-curious robot.

Chapter 4 presents the **third study**, in which a novel robot behavior learning system that models the time-persistent topic of conversation for the purpose of dealing with ambiguous customer questions is presented. A new dataset of human-human interactions in a travel agent scenario was collected to evaluate the system. A novel topic clustering algorithm based on action co-occurrence frequencies and a method for topic estimation during runtime are presented. The proposed technique outperformed several baseline techniques in qualitative and quantitative evaluations. It responded more accurately to ambiguous questions and participants found it was easier to understand, provided more information, and required less effort to interact with.

Finally, Chapter 5 presents the conclusion and directions for future research.

Chapter 2

A Data-Driven Memory Model of Human Actions

Abstract – Many recent studies have shown that behaviors and interaction logic for social robots can be learned automatically from natural examples of human-human interaction by machine learning algorithms, with minimal input from human designers. In this work, we exceed the capabilities of the previous approaches by giving the robot *memory*. In earlier work, the robot's actions were decided based only on a narrow temporal window of the current interaction context. However, human behaviors often depend on more temporally-distant events in the interaction history. Thus, we raise the question of whether (and how) an automated behavior learning system can learn a memory representation of interaction history, within a simulated camera shop scenario. An analysis of the types of *memory-setting* and *memory-dependent* actions that occur in the camera shop scenario is presented. Then, to create more examples of such actions for evaluating a shopkeeper robot behavior learning system, an interaction dataset is simulated. A Gated Recurrent Unit (GRU) neural network architecture is applied in the behavior learning system, which learns a memory representation for performing memory-dependent actions. In an offline evaluation, the GRU system significantly outperformed a without-memory baseline system at generating appropriate memory-dependent actions. Finally, an analysis of the GRU architecture's memory representation is presented.
Keywords: Knowledge representation and reasoning, neural networks, learning from demonstrations, human computer interaction, robotics

2.1 Introduction

Researchers have recently explored the role of social robots in a variety of domains, including elder care [65, 66], shop-keeping [78, 79, 80, 81], hotels [41], education [56], travel [23], and as personal companions [55, 107]. As the capability of artificial intelligence increases, this trend is likely to continue. However, one of the greatest challenges to introducing social robots into new domains is the time-consuming, tedious nature of

designing interaction behaviors for all the scenarios the robot may encounter.

In response to this challenge, previous works have proposed data-driven approaches for designing social robot behaviors automatically, with minimal input from a human designer, by imitation learning from examples of natural human-human interaction [80]. While impressive, with these systems, the robot is limited to responding to contextual information from within only a short temporal window of the overall interaction. But in reality, many interaction behaviors depend on remembering key information over longer time periods.

In general, humans are able to learn a great deal of information about their interaction partners through interaction, which helps them to adapt their behavior to improve the quality of interaction. The information that a person holds in memory about an interaction partner affects how they might explain concepts, describe events, or introduce the interaction partner to others. For example, consider the following interaction between a customer and a shopkeeper:

Shopkeeper: "What brings you to Osaka?"

Customer: "I'm here for work. I often travel between Tokyo and Osaka."

<...Later in the interaction, the *Shopkeeper* introduces a new camera...>

Customer: "Do you have anything else available?"

Shopkeeper: "Since you often travel, the Nikon would suit your needs. It's small and lightweight."

In this example, the shopkeeper is not merely reacting to what the customer said immediately before; he has remembered some information about the customer, which was stated earlier, and made a recommendation based on that memory. Whether or not a robot can learn a memory model of an interaction partner like this purely from example interaction data is an open question. In this work, we propose a data-driven technique that enables a robot to automatically remember such useful information and act on it during interaction with a human.

In our approach, we analyzed the presence of *memory-setting* and *memory-dependent* actions, like those in the above example, in a dataset of human-human interactions in a camera shop scenario. We discovered that there is a wide diversity of these actions, which occur in various patterns of dependence. Since each pattern appears only rarely, it would be difficult for a machine learning algorithm to learn an accurate representation of the shopkeeper's memory. Therefore, we simulated a larger number of interactions that contain memory-setting and memory-dependent actions, and subsequently used it to train and test our proposed learning system. The purpose of the proposed system is to learn the behaviors and interaction logic of the shopkeeper, including how to perform appropriate memory-dependent actions, so that the human shopkeeper can be replaced by a robot. It accomplishes this by using a recurrent neural network architecture with gated recurrent units, which we evaluated in an offline evaluation. Lastly, we present an analysis of how the neural network represents memories.

The research question we sought to answer with this work is *whether* it is possible to build a memory system for human-robot interaction in a data-driven way. To our knowledge, there have not been any previous data-driven, behavior-learning systems

designed and evaluated for the purpose of modeling memories in human-robot interaction. The most similar work to our proposed method are context-sensitive recurrent neural architectures [16, 119]. But, they have only been applied to unimodal interaction data (dialogs), without automatic speech recognition errors, which are common in HRI. Furthermore, we go a step further than training a testing a model (Sec. 2.6 by first analyzing the types of memory-dependent behaviors that occur in human-human interaction and then conducting an analysis to illustrate *how* a gated recurrent neural network represents memories (Sec. 2.7.

2.2 Related works

2.2.1 Memory for Social Robots

There are several fields of study related to modeling memory for human-robot interaction. **Long-term HRI** focuses on designing robots that can build positive relationships with individual humans over time periods encompassing multiple interactions [34, 59, 74]. In such interactions, the robot's *long-term* memories of its previous interactions with individuals it has previously interacted with can help build positive relationships. In contrast, our work focuses on the use of robots in domains where the human and robot will not necessarily meet more than once, e.g. between a shopkeeper and a customer; thus, the primary challenge is to generate robot actions consistent with the context of the current interaction. Therefore, we concentrate on *short-term* memory and not *long-term* memory.

Cognitive architectures are models of human cognition which contain components such as *working memory* and *episodic memory* [67, 126], which are more similar to the *short-term* memory modeled in our approach. Cognitive architectures have also been used to control robots and to predict human actions in HRI [5]. But, this approach relies on hand-coded models based on first principles, which can be difficult to design. In contrast, the data-driven approach to designing robot behaviors and interaction logic can be automated so it requires less effort from designers.

User modeling is used in dialog systems, and more recently HRI, to model users' beliefs, goals, and plans such that the system can adapt to individual users [28, 62, 129]. Typically, user models are manually defined by domain experts, i.e. *what* is to be remembered by the system must be specified prior to interaction. Conversely, our memory modeling approach is data-driven, wherein the contents of memory are learned automatically, without input from expensive domain experts.

2.2.2 Machine learning approaches to memory

An agent's decision of which action it should take given some interaction context is often formulated as a sequential decision making process. Recurrent neural networks (RNN) are a data-driven method for modeling sequential processes, where, for example, the input to the network can be an interaction history (a sequence of human and

agent actions), and the output could be the agent's next action [77]. Long Short Term Memories (LSTM), and the simplified variant Gated Recurrent Units (GRU), are more advanced versions of RNNs designed to 'remember' important information for longer durations [19, 48]. Memory Networks, Dynamic Memory Networks, and Neural Turing Machines are three newer neural architectures for sequence modeling that store past inputs in a searchable memory buffer [37, 64, 121]. Our proposed system implements a GRU to decide the robot's actions.

2.2.3 Data-driven learning of interaction behaviors for social robots and virtual humans

Liu et al. introduced the concept of training a social robot from examples of natural human-human interaction without any input from a human designer [78, 79, 80, 81]. The approach consists of clustering the raw interaction data to find discrete common actions, and then training a supervised learner using the action cluster IDs as labels. These systems have proven to be effective, but the robot's next action is decided based on only a narrow temporal window of the current interaction state.

Admoni and Scassellati introduced a system that uses supervised learning with hand-labeled data to recognize and generate non-verbal behaviors for a robot tutor [1]. Leite et al. introduced a 'semi-supervised learning' game-playing robot that could query workers via Amazon Mechanical Turk for new lines of dialog whenever it anticipated a novel situation [75]. These methods rely on hand annotated data or input from human designers, so they are not appropriate for rapid development of robot behaviors to new domains.

There also exist data-driven systems for designing behaviors for virtual humans. The 'Restaurant Game' was a virtual restaurant in which two human players, playing the roles of customer and waiter, could navigate around the restaurant and interact via text and predefined actions [92, 93]. 'Plan networks' were automatically extracted from the interaction data, enabling the automation of each character. The 'Mars Escape' game used a similar approach, and went a step further by embodying the learned behaviors in a real robot [9, 17, 18]. In contrast, data collected in the real, physical world, which is incorporated into our interaction simulation, has the additional challenge of sensor noise, which is absent in virtual worlds.

2.2.4 Data-driven dialog systems

Data-driven approaches are also popular for training dialog systems, where RNNs (and variations such as LSTM [48] and GRU [19]) have become a common method for context-sensitive language generation [112, 119]. Tian et al. compared several neural network architectures that integrate dialog context for response generation and found that hierarchical sequence models with attention performed the best [124]. Furthermore, their results showed that models with context were able to generate longer, more meaningful and diverse replies than models without.

In the recent Dialog Systems Technology Challenge several methods were presented for autonomously learning dialog behaviors from unannotated interaction datasets of Twitter interactions and simulated restaurant recommendation interactions [95]. The most successful approaches used variants of RNNs with LSTM. Similarly, other RNN architectures for unsupervised, end-to-end learning of context-sensitive conversational models demonstrated on Twitter data have been proposed [119].

Our approach is inspired by the related works on data-driven dialog systems, but there are some differences. Their training and testing data are not multimodal and do not contain speech recognition errors, in contrast to our data. Furthermore, little analysis is offered of why LSTMs work so well in these related works, but we offer an analysis of memory in interaction that motivates our network architecture.

The FRAMES corpus consists of manually annotated dialog transcripts in the travel agent domain collected in a Wizard-of-Oz setting and was designed for "adding memory to goal-oriented dialog systems" [26]. Shultz et al. present a system trained on this dataset that can remember which travel options were discussed so the agent can aid the customer in comparing options [111]. In contrast, in our work, we propose a system that learns to remember useful information without manually annotated labels.

Janarthanam and Lemon presented a finite-state based dialog system for setting up home broadband networks that learned to adapt to users with different levels of knowledge using reinforcement learning [52]. This is similar to our system in that it can remember customer information and uses to to decide actions. However, in their work the user knowledge levels and system actions were manually predefined. In contrast, we present a system that automatically learns from interaction data the set of system actions, the action policy, and which information is relevant for adapting to the user.

2.3 Analysis of human-human interaction data

2.3.1 The human-human interaction dataset

The human-human interaction dataset from [80] was used to discover how memory was used in interaction, and to inspire the design of an interaction simulator.

2.3.1.1 Scenario

The scenario consisted of typical interactions in a camera shop domain. A camera shop environment was set up in an 8m x 11m experiment space containing a service counter and three camera models (a Canon EOS 5D Mark III, a Sony Alpha a6000, and a Nikon Coolpix S2800) in different locations, as shown in Fig. 2.1. Each interaction contained one shopkeeper participant and one customer participant.

Note, this camera shop scenario was intentionally simplified in comparison to a real camera shop interaction, which would contain many more than three cameras. In a real camera shop scenario there are numerous interaction patterns, and multiple examples of each pattern would be required for a system to learn appropriate robot behaviors.

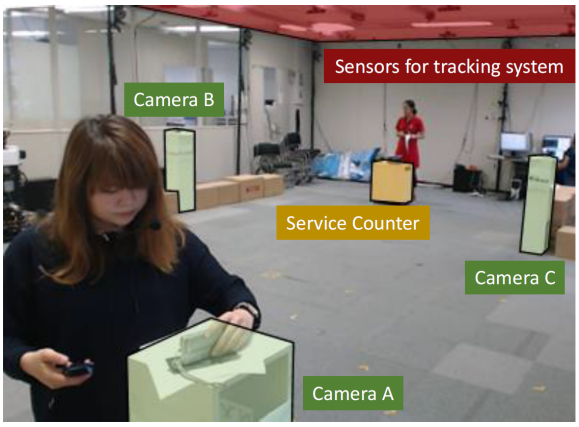


Figure 2.1: The camera shop setup contained three camera displays. Sensors on the ceiling were used for tracking human position, and smartphones carried by the participants were used to capture speech [78].

Therefore, it was necessary to constrain the interaction scenario to collect a sufficient quantity of data for training and testing the system. In the future it will be possible to passively collect large quantities of natural, human-human interaction data in the real world, but for now behavior learning systems must be trained and tested in simplified scenarios.

2.3.1.2 Sensors

The participants' speech and position data was recorded automatically as they interacted with each other. A sensor network with a human position tracking system recorded the participants' location data [11].

To collect speech data, participants were required to use a handheld smartphone with an attached head-mounted microphone connected to the Google Speech API. To start and stop recording their speech a participant could touch anywhere on the smartphone's screen and an audible cue would play. This setup enabled participants to speak and operate the smartphone without distraction.

2.3.1.3 Participants

A total of 18 fluent English speaking participants, with varied levels of knowledge about cameras, were recruited to play the role of the customer. In order to obtain a diverse set of behaviors, two different participants, with different interaction styles, were chosen to play the role of the shopkeeper.

2.3.1.4 Procedure

The participants were encouraged to act naturally and to focus discussion on the features listed on camera spec. sheets (8 to 10 features per camera). Customer participants

were encouraged to play with the cameras, browse the shop, or ask camera-related questions. The shopkeeper participants were instructed to wait at the service counter at the start of the interaction, be polite, and behave according to their role (e.g. greetings and farewells, letting the customer browse, answering questions, or introducing products when appropriate).

2.3.1.5 Summary of collected data

In total, 405 interactions were collected, containing 4966 turns. Moreover, the data contained 4061 shopkeeper utterances and 4115 customer utterances. Several natural variations of phrasing (e.g. backchannels such as, "okay" and "I see, and...") and terminology are present in the dataset [80]. Therefore we consider the behaviors in the dataset to be realistic.

The dataset contained many automatic speech recognition errors. Of 400 utterances from a dataset collected in the same environment and scenario, and using the same equipment and recording methods, 53% were correctly recognized, 30% had minor errors (e.g. "can it should video" instead of "can it shoot video"), and 17% were complete nonsense [78]. The presence of sensor errors is one of the challenges of data-driven HRI; therefore, these ASR errors remain in the datasets.

2.3.2 Analysis of memory-dependent actions

To discover how memories of events during interaction were used by the participants, we analyzed the actions in the human-human dataset.

2.3.2.1 Defining memory-dependent and memory-setting actions

Interactions in the dataset were discretized into sequences of alternating customer and shopkeeper *actions* [80]. Each *action* has a speech component (the participant's utterance) and a spatial state component (the participant's current location, movement departure location, movement arrival location, and a joint spatial formation – *waiting*, *face-to-face*, or *present object*). When exploring how memory was used in these interactions it helps to distinguish two special types of actions.

Some shopkeeper actions depend heavily on actions that occurred earlier in the interaction. A shopkeeper action that depends on an earlier action is called a ***memory-dependent*** action. The earlier action on which a memory-dependent action depends is called a ***memory-setting*** action. For example, if in the beginning of an interaction a customer states that they often travel, "I'm here for work. I often travel between Tokyo and Osaka" (a memory-setting action), then the shopkeeper should remember this. Then, later in the interaction the shopkeeper can make a recommendation, "Since you often travel, the Nikon would suit your needs. It's small and lightweight." (a memory-dependent action). Throughout this work, we use the abbreviations *mem-dep* and *mem-set* for memory-dependent and memory-setting actions, respectively.

Mem-Dep type	Type of memory Mem-Dep depends on	Example Mem-Set utterance	Example Mem-Dep utterance
Knowledge	Customer's level of knowledge	"I don't really know much about cameras."	"It has automatic settings so you don't have to adjust anything."
Picture type	What customer wants to take pictures of	"I mostly take pictures at sporting events."	"For sport pictures it's better because it has 179 autofocus points, so it's really good for taking moving objects."
Feature	Customer's desired feature	I'm looking for something cheap"	"It's a bit out of your budget this one is $550"
Topic continue	Continuation of previous topic	"This one has full manual settings?"	"But it also comes with manual control"
Additional info.	Which information has already been presented	<shopkeeper has not yet introduced the camera's weight>	"It is also very light"
Previous camera	Previous camera of conversation	<shopkeeper was previously talking about the camera>	"The previous camera has a red color so I recommend that for you."
Non-repeating	Action is not repeated	<shopkeeper has not yet greeted customer>	"Good afternoon how can I help you?"
Other	Other	"I'm looking for a present for my father. He's not very good with technology though."	"Just remember, when thinking about your father, with this camera just point and shoot. You can use this right away."

Table 2.1: Types of memory-dependent actions discovered from the human-human dataset

2.3.2.2 Types of memory-dependent actions

We identified 8 types of memory-dependent shopkeeper actions in the human-human dataset. These are shown in Table 2.1. The leftmost column of Table 2.1 contains the names of the mem-dep action types; the second column contains the type of information in memory that each mem-dep action type depends on; the third column contains example mem-set utterances, which trigger the relevant information to be stored in memory; and the last column contains example sentences of each mem-dep action type.

We discovered memory-dependent actions that depend on the customer's level of knowledge (*knowledge*), photography subject (*picture type*), and preferred camera features (*feature*). We also discovered mem-dep actions that continue the previous topic of conversation (*topic continue*). Some mem-dep actions do not simply depend on memory of a single action. Actions in which the shopkeeper introduces a camera or feature (*additional info.*) depend on memory of which cameras and features have already been talked about. Similarly, some mem-dep actions refer back to cameras that were previously under discussion (*previous camera*). Mem-dep actions that should not be repeated (*non-repeating*), such as greetings, depend on memory of whether the action has already been performed. Finally, some mem-dep action depended on idiosyncratic characteristics of the participants or interaction (*other*).

2.3.2.3 Memory-dependent action statistics

After identifying the common mem-dep action types, we annotated each shopkeeper action with 1) the mem-dep type and 2) the index of the mem-set action on which it depended. We annotated 300 of the human-human interactions, which contained 3666

Mem-Dep type	Number of occurrences	Percent of the dataset	Ave. num. turns from Mem-Set (and s.d.)
Knowledge	23	0.6%	4.1 (3.8)
Picture type	54	1.5%	3.5 (3.7)
Feature	24	0.7%	1.4 (1.9)
Topic continue	272	7.4%	0.5 (2.1)
Additional info.	66	1.8%	2.1 (2.4)
Previous camera	56	1.5%	4.7 (3.6)
Non-repeating	192	5.2%	N/A
Other	19	0.5%	3.8 (4.0)
Total	706	19.2%	

Table 2.2: Occurrence frequency of memory-dependent action types and distance from the corresponding memory-setting actions. The numbers were computed from a subset of 300 human-human interactions containing 3666 turns.

turns. Table 2.2 contains the important information discovered from annotation.

The second and third columns from the left in Table 2.2 show the number of occurrences of each mem-dep type and the percent of the 3666 annotated turns that are that type, respectively. With the exceptions of *topic continue* (7.4 % of the data) and *non-repeating* (5.2% of the data), the mem-dep actions occur infrequently – the remaining 6 mem-dep types combined only constitute 6.6% percent of the data. The low number of examples suggests that it may be difficult for machine learning algorithms to learn to generate these types of actions.

The distance between a memory-setting action and a corresponding memory-dependent action depends on the memory-dependent action type. The last two columns display the average number of turns between each mem-dep action type and the mem-set actions on which they depend (and standard deviation). Note that the mem-dep types *knowledge* (average of 4.1 turns from mem-set, s.d. 3.8), *picture type* (ave. 3.5, s.d. 3.7), *previous camera* (ave. 4.7, s.d. 3.6), and *other* (ave. 3.8, s.d. 4.0) typically occur more than three turns after the mem-set action. In contrast, *topic continue* (ave. 0.5, s.d. 2.1), *feature* (ave. 1.4, s.d. 1.9), and *additional info.* (ave. 2.1, s.d. 2.4) usually depend on much more local information. Since previous techniques have explored incorporation of local context in robot action prediction [79], in this work we focus on actions like the former, that have long-term dependencies.

2.4 Interaction simulation

The focus of this work is to discover whether a robot is capable of learning to represent memories so that it can correctly perform actions that depend on those memories; however, the number of training examples in the human-human dataset is lacking. The

human-human camera shop dataset described above contains a diverse set of memory-setting and memory-dependent actions, which appear in various different patterns of dependence. Furthermore, each type of memory-setting and memory-dependent action can occur with many natural variations of utterance, and may contain automatic speech recognition errors. Consequently, the number of examples of each pattern is lacking, and thus difficult for a machine learning algorithm to learn.

We believe that in the future it is likely that much larger datasets of natural human-human interactions will be collected by sensor networks deployed in spaces where humans frequently interact. But in the meantime, we have developed an interaction simulator based on the interaction patterns observed in the human-human data, to create a larger training dataset with more examples of memory-setting and memory-dependent actions.

2.4.1 Procedure

The simulator works by generating sequences of alternating shopkeeper and customer actions (Table 2.7 in the appendix). Moreover, in each generated interaction, the simulator uses action transition probabilities in combination with if-then style logic to generate contextually appropriate actions. Finally, customer and shopkeeper utterances from the human-human dataset are injected into the generated action sequences.

2.4.1.1 Action transitions

Each simulated interaction starts with the customer at the door and the shopkeeper at the service counter. The customer then enters and approaches one of the three cameras or the shopkeeper. The shopkeeper may approach a browsing customer to assist them, by asking their preferences, answering questions, or introducing the cameras and their features. In some cases the customer may state they simply wish to browse. Otherwise, the customer may ask about the cameras or state their preferences. The interaction proceeds until the customer decides to leave the store. Throughout the interaction, the customer and shopkeeper's actions are partially decided by action transition probabilities.

The action transition probabilities were set manually based the interactions in the human-human dataset from [80]. In that work, the customer's and shopkeeper's actions were segmented and discretized into sequences of alternating customer and shopkeeper actions. The transition probabilities were estimated based on observations of these sequences. For example, after the shopkeeper introduces a camera, there is a roughly 50% chance that the customer will remain silent (allowing the shopkeeper to take initiative) and a 50% chance that the customer will ask a question about a feature. In the future, it would be useful to compute the transition probabilities automatically by first clustering the human actions based on similarity (as in Sec. 2.5.3 and then counting the transitions between the *action clusters*.

2.4.1.2 If-then logic for contextually appropriate actions

For each interaction, the simulator modeled important information about the interaction history, including: whether the customer has stated they are just browsing and wish to be left alone, the current camera of conversation, a list of memory-setting actions that have been performed, a list of previous cameras of conversation, a list of previous features of conversation, and the customer and shopkeeper's locations.

Using if-then logic conditioned on these stored aspects of the interaction history, the simulator generates contextually appropriate actions. For example, the shopkeeper's actions are simulated such that he does not repeatedly approach browsing customers who wish to be left alone, remembers the preferences of the customer and makes appropriate recommendations, and remembers which cameras and features have already been talked about so that he does not re-introduce them. Additionally, the customers' actions are simulated such that they do not ask about features or cameras that have already been introduced or asked about

To generate memory-setting actions customized for different *customer types*, if-then logic was used. We designated 10 possible customer types based on the types of pictures they like to take and their level of expertise: *art school*, *simple everyday pictures*, *family and friends*, *nighttime*, *outdoor*, *pets*, *sports*, *travel*, *expert user*, and *novice user*. Whenever the customer performed a LOOKING_FOR_A_CAMERA_WITH_X action (from Table 2.7), X was determined based on the customer type, resulting in a customized memory-setting action.

To generate customized memory-dependent actions, if-then logic conditioned on the occurrence of the above memory-setting actions was used. Moreover, each customer type was associated with preferred camera features (Table 2.8 in the appendix). Using this knowledge, the shopkeeper was able to perform customized memory-dependent actions by appending phrases relating to the customer type to responses about their preferred features (e.g. "This one is very lightweight, *which makes it suitable for travel*."). (The cameras' features are shown in Table 2.9 of the appendix.)

2.4.1.3 Natural variation and speech recognition errors

In the final step of simulation, customer and shopkeeper utterances are assigned to each turn. For each action (from Table 2.7), a set of matching utterances was selected from the human-human dataset. The utterances from the human-human dataset contain natural variation in phrasing and also contain many speech recognition errors. By introducing these characteristics into the simulated dataset, it should present a similar learning challenge as the real human-human dataset.

Time Step	Person	Location	Utterance	Camera of conversation	Previous cameras and features of conversation	Memory	Action Type
1	C	DOOR					
	S	SERVICE COUNTER					
2	C	SONY					
	S	SONY	good afternoon how can I help today				
3	C	SONY	what cameras do you have help with that would be good for beginners			NOVICE	Mem-Set 'Novice user'
	S	NIKON	this kind of camera over here is fantastic it's like the next step along for using your mobile phone	NIKON	NIKON	"	
4	C	NIKON	okay let's see	NIKON	"	"	
	S	NIKON	this one is a point and shoot camera perfect for beginner user, you can just push the shutter button and you don't have to adjust anything it will auto adjust itself	NIKON	NIKON: point_and_shoot	"	Mem-Dep 'Novice user'
5	C	NIKON	huh	NIKON	"	"	
	S	NIKON	this camera is really good, it is only $68, it's incredibly light, feel it up pick it up feel the weights	NIKON	NIKON: point_and_shoot, price	"	
			⋮				
9	C	NIKON	okay thanks for the information		NIKON: point_and_shoot, price, ease_of_use, color, …	"	
	S	NIKON	great thanks for your time have a great afternoon		"	"	
10	C	DOOR			"	"	
	S	SERVICE COUNTER			"	"	

Table 2.3: An example simulated interaction

2.4.2 The simulated dataset

In total, we simulated 50,000 interactions, containing 619,848 customer-shopkeeper turns, for training and testing the proposed system. The data is available for download[1]. Table 2.3 presents an example simulated interaction.

[1] http://www.geminoid.jp/dataset/simulatedcamerashop/Dataset%20-%20Simulated%20camera%20shop.html

2.5 The data-driven memory learning system

2.5.1 Overview

The goal of the system is to learn the behaviors of a shopkeeper from examples of human-human interaction, such that a robot can perform as a shopkeeper in interaction with real humans. Fig. 2.2 shows an overview of the complete system that enables the robot shopkeeper to function.

A sensor network collects raw interaction data in the camera shop. It consists of an array of Microsoft Kinect sensors that track the human and robot's positions and a smartphone application that records the customer's speech. Automatic speech recognition is used to transcribe the customer's speech into text. The sensor data is sent to a behavior abstraction module that discretizes the raw tracking data into one of the typical stopping locations (door, middle, Canon, Sony, Nikon, and service counter), which were discovered by clustering the stopping locations in the training interactions [78].

The interaction history stores all the customer and shopkeeper's previous actions (from turn 0 through t-1). When a new customer action is detected, it is appended to the interaction history. The actions are vectorized (Sec. 2.5.2 and fed into a neural network which outputs the robot's next action (Sec. 2.5.4, which also gets appended to the interaction history. In this way, a robot can perform the role of a shopkeeper in live interaction with a human.

In this work, we focused on the role of the GRU neural network in representing memories and enabling the robot to perform appropriate memory-dependent actions. Therefore, we trained the system on simulated interactions and tested it in an offline evaluation, rather than conducting live user studies. Similar versions of the complete system have been demonstrated to work in live interactions with real humans in previous work [78, 79].

2.5.2 Input action vectorization

The sequence of actions in an interaction must be vectorized before it can be input into the neural network. The input vectorization for a customer or shopkeeper action consists of a speech vector concatenated to a location vector.

Speech was represented using a binary bag-of-words representation, where each index of the speech vector represented a word in a vocabulary, with a value of 1 if the word occurred in the utterance, and 0 otherwise. Separate vocabularies were acquired for the customer and shopkeeper based on words spoken by customers and shopkeepers in the simulated dataset. The customer and shopkeeper vocabularies contained 938 and 1255 words, respectively.

We opted for this simple speech representation for the purpose of reducing the input dimensionality (and thus the number of neural network parameters and training time) and increasing possibilities for interpretability. One limitation of bag-of-words representation is that word ordering information is lost. Previous works [80] used n-gram

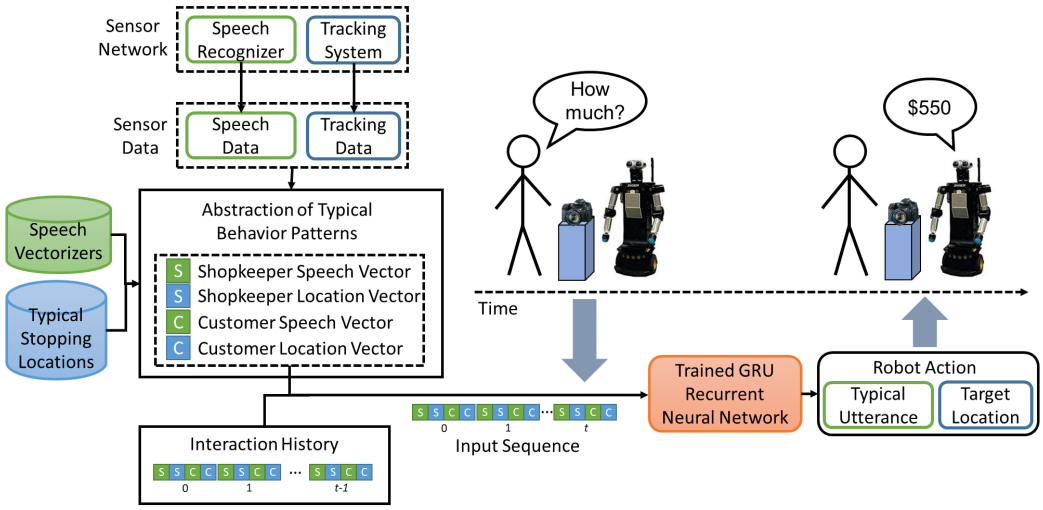


Figure 2.2: Overview of the complete system for real-time interaction

vectorization (n=1, 2, and 3) with latent semantic analysis (LSA) for dimensionality reduction. However, LSA works by combining multiple n-grams into the same dimension, so interpretability is lost.

The location vector represented at which of the 6 locations (door, middle, Canon, Sony, Nikon, and service counter) the customer/shopkeeper was located. For these we used a one-hot vector of 5 dimensions (since the customer never goes to the service counter and the shopkeeper never goes to the door).

The input vector for a single time step t consisted of the shopkeeper action vector s_{t-1} (1255 speech and 5 location dimensions), and the customer action vector c_t (938 speech and 5 location dimensions) concatenated together, thus a vector of 2203 dimensions.

2.5.3 Output shopkeeper action clustering

The neural network works as a 'sequence classifier', which learns to output the most appropriate 'class' for any input 'sequence'. The output classes dictate the robot's next action. To obtain these classes, the simulated shopkeeper actions were clustered into clusters of similar actions, representing the set of common shopkeeper actions. To discretize the shopkeeper actions, each one was assigned the ID of its containing cluster. Shopkeeper actions were clustered by combining shopkeeper speech clusters with shopkeeper location information.

The goal of speech clustering was to find highly homogeneous clusters of semantically similar utterances. To do this, 25,000 utterances were sampled from the simulated interactions, speech vectors and inter-utterance cosine distances were computed, and the Dynamic Tree Cut hierarchical clustering algorithm (DTC) was applied [70]. The clustering procedure yielded 1083 speech clusters.

The DTC algorithm has several advantages for speech clustering. We experimented with several clustering algorithms and found DTC to consistently yield large, homogeneous clusters of semantically similar utterances. Furthermore, DTC automatically discovers the number of speech clusters from the data, which helps to automate the robot behavior learning process. Also, in contrast to traditional hierarchical text clustering methods [2], it cuts the tree structure at different heights depending on the degree of variation of utterances in the branches. This is suitable for speech data since semantically similar utterances can have a wide range of variation depending on utterance length, natural variation of phrasing, and presence of ASR errors.

For the purpose of speech clustering, a separate speech vectorization model was used, which encoded word ordering and emphasized important keywords. Each speech vector consisted of an n-gram vector (n=1, 2, and 3) concatenated to a keyword vector. Keywords were extracted by a cloud-based API (now part of IBM Watson[2]).

Speech clusters were combined with location information to yield the action clusters. More specifically, to get the shopkeeper's action cluster ID at t, the ID of the speech cluster containing the shopkeeper's utterance at t is combined with his current location (service counter, middle, Sony, Canon, or Nikon), yielding an action cluster ID. For example, if the shopkeeper says "Thank you for coming", which is in speech cluster *3*, while standing near the *Sony*, then the action cluster ID is *3_Sony*. Combining the speech clusters with the shopkeeper location information yielded 1479 shopkeeper action clusters.

Since the output robot action class must dictate the robot's speech, a *typical utterance* was automatically selected for each speech cluster by finding the utterance that was most similar to all the other utterances in the speech cluster (medoid). The Levenshtein distance metric normalized for the length of the utterances was used. Complete utterances with few ASR errors tended to share the most similarities with other utterances in the same cluster. Therefore, *typical utterances* tended to be well formed and easy to understand.

2.5.4 Recurrent neural network architecture

The GRU recurrent neural network takes a vectorized input sequence $s_0, c_1, \ldots, s_{t-1}, c_t$ as input and outputs a probability distribution over the possible next shopkeeper actions, s_t. The neural network architecture consists of six layers, as shown in Fig. 2.3a. The first layer is the input layer, which takes as input each of the $s_{t'-1}, c_{t'}$ vectorizations, for $0 < t' \leq t$ (that is, all actions from the beginning of the interaction up through the most recent customer action). The next three layers are dense hidden layers of 800 units each, with leaky rectifier activation functions, whose purpose is to condense the input into a lower dimensional encoding. The next layer is the gated recurrent unit (GRU) layer, as defined in [19]. Finally, the last layer is the output layer, which represents the robot's next action.

[2] https://www.ibm.com/watson/services/natural-language-understanding/

The GRU layer is the most critical part of the architecture for learning memories of important events in the interaction. It does this by maintaining a state vector, h_t, over multiple time steps. The goal of training the network is for the GRU to learn to represent the occurrence of memory-setting actions within its state vector, such that the network can output appropriate memory-dependent actions.

The behavior of the GRU layer is defined with the following four equations:

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t] + b_z\right) \tag{2.5.1}$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t] + b_r\right) \tag{2.5.2}$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t] + b\right) \tag{2.5.3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t, \tag{2.5.4}$$

where z_t is the update gate output, r_t is the reset gate output, $\tilde{h}_t$ is the candidate GRU state, h_t is the new GRU state (and output), and x_t is the current input (the output of the third dense hidden layer). The goal of r_t is to learn when to add information from the previous state h_{t-1} to the candidate state $\tilde{h}_t$ vs. when to only add information from the current input x_t to the candidate state $\tilde{h}_t$. The goal of z_t is to learn when to keep the previous state h_{t-1} vs. when to overwrite it with the candidate state $\tilde{h}_t$. W_z W_r, W, b_z, b_z, and b are weight and bias parameters that are learned during training. In total, the weights and biases constitute 8,071,879 parameters.

2.6 Offline system evaluation

To understand the role of the GRU in representing memories, we conducted an offline evaluation comparing the proposed data-driven memory system to a system without a GRU layer (i.e. memory). By comparing the proposed system to a baseline that is the same in all resptects except for the lack of a GRU layer, the effects of the GRU layer on the system's performance can be isolated.

2.6.1 Experimental conditions

The focus of this study was to develop a technique for generating appropriate memory-dependent actions by remembering relevant memory-setting actions that occur during interaction. Therefore, the experimental design compared the behaviors of two action prediction conditions.

No memory perceptron (NMP) consisted of a five layer, feedforward neural network. The first layer is an input layer that takes the vectorization of the most recent customer action, the next three layers are dense hidden layers with 800 dimensions, and the last layer is a softmax layer of 1479 dimensions, representing a probability distribution over the possible next shopkeeper actions. This architecture is similar to the proposed memory system, but without the GRU layer. Thus, it cannot 'remember' past

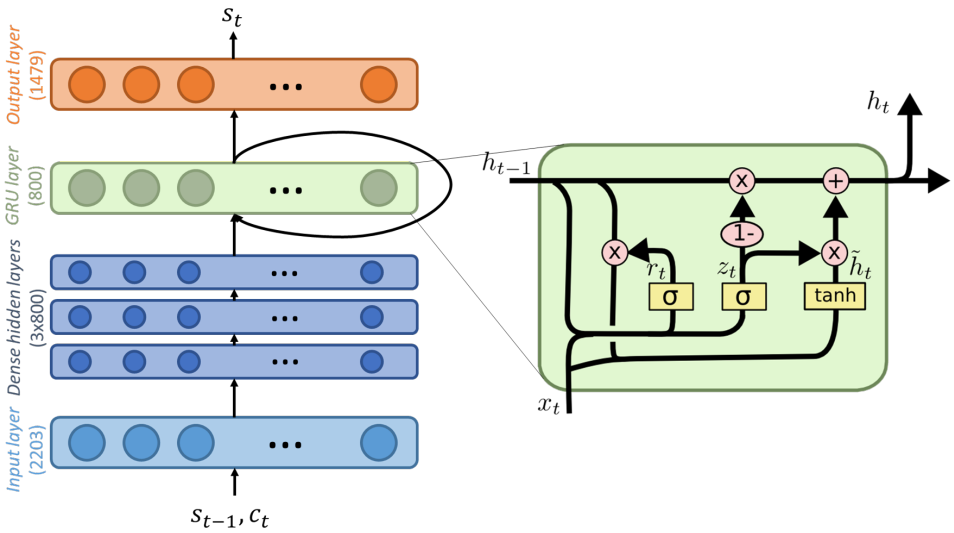


Figure 2.3: a) The neural network architecture: s_{t-1} is the previous shopkeeper action (1255 speech and 5 location dimensions), c_t is the current customer action (938 speech and 5 location dimensions), and s_t is the next shopkeeper action (1479 shopkeeper action clusters), which dictates the robot's next action. b) The internal details of the GRU layer **(Olah, 2017)**: h_{t-1} and h_t are the previous and current GRU states, r_t is the reset gate output, z_t is the update gate output, and $\tilde{h}_{t-1}$ is the candidate GRU state.

information. By comparing the proposed system to the NMP we can discover how the GRU layer, and thus 'memory', affects performance.

Proposed system consisted of the complete system introduced in the previous section.

2.6.2 Training the systems

The two action predictors were trained using the same procedure. The first 75% of the simulated dataset (containing 464,612 customer input / shopkeeper output interaction turns) was used for training and the first 500 turns (39 interactions) from the remaining 25% was used for testing. Categorical cross-entropy was used to compute the loss function. To encourage the networks to output memory-dependent actions, which occurred infrequently in the training data, the loss for each output action was weighted inversely proportional to the action's number of occurrences in the training data. The Adam optimizer was used to update the parameter values over 200 epochs [60].

2.6.3 Evaluation procedure

One human evaluator, an internally-recruited fluent English speaker (F, age 37) blind to the experimental conditions, evaluated each predicted robot action with a binary label of either *correct* or *incorrect*. To receive a *correct* label the robot's action must have provided the correct information. If the customer did not request any specific information, then the criterion was that the robot's action must be socially appropriate. In the case that the evaluator was unable to determine if the predicted action was correct or not, the instance was not included in the evaluation (0.2% of instances). Evaluations by a second evaluator, an internally-recruited fluent English speaker (M, age 28) blind to the experimental conditions, showed a high degree of agreement, with a kappa coefficient of 0.63, so the evaluations were judged to be reliable.

To better understand the role of memory in outputting memory-dependent actions, we narrowed our focus to the *memory-dependent* actions that depend on *customer type* (described in Sec. 2.4.1.2). The 500 action predictions were divided into *customer-type-memory-dependent* and *other* subsets. If a robot action contained an utterance from a simulated action that depended on memory of customer type, it was labeled *customer-type-memory-dependent*. Otherwise, it was labeled *other*.

2.6.4 Results

The evaluation results and statistical analysis results are shown in Table 2.4. Overall the proposed system performed the best with 85.8% action predictions judged correct. This was significantly better than the no memory perceptron (NMP) with 70.3% correct (χ^2(4.4.1)=34.68, $p<.001$).

On the *customer-type-memory-dependent* subset of action predictions, the proposed system had 66.7% action predictions correct. This was significantly better than the NMP with 13.6% correct (χ^2(4.4.1)=16.26, $p<.001$).

On the *other* subset of action predictions, the proposed system had 87.5% correct. This was significantly better than the NMP with 73.0% correct (χ^2(4.4.1)=31.07, $p<.001$).

Lastly, the proposed system output almost twice as many *customer-type-memory-dependent* actions as the NMP (42 actions vs. 22 actions, respectively) (χ^2(4.4.1)=6.68, $p=.010$)."

2.6.5 Analysis of incorrect actions

We analyzed the proposed system's 14 incorrect *customer-type-memory-dependent* actions to understand the cause of the system's mistakes. 36% were incorrect because of inappropriate repetition, 36% because the robot did not provide the requested information, 14% because the robot's speech did not match the customer type (e.g. recommending a camera for simple, everyday photos to someone who wants to do outdoor photography), and 14% because the system did not distinguish between customer subtypes (e.g. customers who owned cats vs. those who owned dogs).

Condition	Overall Correct	Customer-type-mem-dep Correct	Other Correct
No Memory Perceptron	70.3% (N=499)	13.6% (N=22)	73.0% (N=477)
Proposed System	**85.8%** (N=499)	**66.7%** (N=42)	**87.5%** (N=457)
Chi Square Test	$p<.001$., $\chi^2=34.68$	$p<.001$, $\chi^2=16.26$	$p<.001$, $\chi^2=31.07$

Table 2.4: Percent of the Proposed and No Memory Perceptron systems' actions judged correct by a human evaluator in the offline evaluation. Customer-type-mem-dep refers to the subset of output actions that depend on memory of customer type. Other refers to the remaining output actions. Overall is the both of these subsets combined. N refers to the number of actions that were evaluated for each system/subset. The last row contains the Chi Square test results comparing the two systems on each action subset. The proposed system performed significantly better on all action subsets.

Furthermore, analysis revealed that the proposed system inappropriately repeated utterances less frequently that the NMP. 1.4% (7 instances) of the proposed system's actions were inappropriately repeated while 4.4% (22 instances) of the NMP's were inappropriately repeated ($\chi^2(4.4.1)=7.99$, $p=.004$). This demonstrates the importance of memory in avoiding repetition.

2.6.6 Discussion

2.6.6.1 The importance of memory

The proposed system had some ability to 'remember' the interaction history: it learned to correctly output *customer-type-memory-dependent* actions with 66.7% accuracy. On the other hand, the no memory perceptron (NMP) only output *customer-type-memory-dependent* actions correctly with 13.6% accuracy, since it had no knowledge of which memory-setting actions had occurred. This demonstrates the value of learning a memory representation for human-robot interaction.

The proposed system also performed better than NMP on the *other* subset of actions (87.5% vs. 73.0%). We observed that many of the NMP's mistakes resulted from 1) the robot repeatedly asking the customer what their preferences were, 2) re-greeting a customer who stated they want to browse and be left alone, and 3) reintroducing camera features that the robot had already introduced. To behave appropriately in these situations the robot needs to remember the interaction history.

2.6.6.2 Generalization and overfitting

The proposed system has more learned parameters than training data, but this did not result in overfitting. The proposed system has about 17.4 times as many parameters as training data (8,071,879 parameters and 464,612 training points), but this is common practice with neural networks. Related work using neural networks in domains conceptually related to robot behavior learning, including machine translation ; dialog

modeling [119]; and question answering [121, 132], present neural networks with in the order of 10 and 100 times as many parameters as training data.

The proposed system was able to generalize from the training set to the testing set, as demonstrated by the proposed system's high accuracy on the testing dataset (85.8% correct overall). To further enhance this point, here we show that the simulated testing interactions were distinct from the simulated training interactions.

Analysis showed that the 39 testing interactions were indeed distinct from the training interactions. The normalized Levenshtein distance metric (NLD) was used to compute the differences between the 39 testing interactions and the training interactions. To compute the NLD between a testing interaction and a training interaction, first each unique combination of utterance and location was assigned a unique ID. Then, each interaction was represented as a sequence of these IDs. Finally, the NLD was computed between the two ID sequences. The average, minimum, and maximum distances between a testing interaction and a training interaction were 0.83 (s.d. 0.07), 0.33, and 0.97, respectively. Thus, the testing interactions were distinct from the training interactions. In conclusion, the proposed system generalizes without overfitting.

2.7 Analysis of the trained neural network

An analysis of the trained GRU model was conducted to elucidate how it represented memories in the camera shop scenario. The GRU layer of the neural network consisted of 800 units like that shown in Fig. 2.3b. Each of these units were trained to store relevant information from the input customer and shopkeeper actions over multiple time steps such that the network could output the most appropriate shopkeeper actions. The goal of the analysis is to examine the behavior in the GRU units that represent 'memories'. That is, which customer / shopkeeper actions are stored in these GRU units? And, what is the behavior of the GRU gates and output activations of these units?

First, the GRU units that are most likely to encode memories are found by computing their mutual information with ground truth memory labels. Second, we explain how treating GRU gate output values as binary values yields a simplified interpretation of their behavior that can be easily visualized. Lastly, the first two steps are combined to visualize the behavior of the memory encoding GRU units over time in two of the simulated interactions.

2.7.1 Finding GRU dimensions that encode memories

To decide which of the 800 GRU units would be most interesting to examine in detail, the mutual information scores (MI) between ground truth memory labels and GRU units were computed from the training data [22]. MI aids in understanding the statistical relationships between certain GRU units and memory types based on aggregated data. The intuitive interpretation of MI is the amount of information given about one variable

Memory Label	Art school	Everyday photos	Expert user	Family and friends	Nighttime	Novice user	Outdoor	Pets	Sports	Travel
GRU unit ID	305	77	601	45	517	524	218	152	363	462
MI (bits)	0.116	0.072	0.099	0.080	0.089	0.057	0.048	0.061	0.051	0.061

Table 2.5: GRU units that give the maximum amount of mutual information (MI) for each memory type. MI aids in understanding the statistical relationships between the GRU units and memory types based on aggregated data. Intuitively, MI indicates the amount of information that knowing the GRU state activation value gives about the occurrence of a memory label (and vice-versa).

(e.g. the occurrence of a memory label) when some other variable is known (e.g. a GRU's activation value). Here, we measure MI in bits. At least 1 bit of information is necessary to know whether a memory-setting action occurred (since it has either occurred or it has not). GRU units that have high mutual information with memory labels are critical for encoding those memories. We refer to these as 'memory units'.

Typically MI is computed between two discrete variables; however, in the case of computing MI between memory types and GRU activations, the former variable is discrete and the latter is continuous. Therefore, the method for computing MI between discrete and continuous variables from [103] was employed, with the parameter k set to 3. This method works by using a k-nearest neighbors based algorithm to compute a probability density function of the continuous variable and then computes MI based on that.

The ground truth memory labels were obtained from the interaction simulator. There were 10 possible memory labels in total, corresponding to the 10 possible *customer types* (knowledge level, preferred subject of photography, etc.) (Table 2.8). To generate the labels, all 10 memory labels were initialized to 0 in the beginning of each interaction. When the customer performed a memory-setting action, e.g. stated their preferred picture type, the appropriate memory label was set to 1 for the remaining time steps in the interaction. In this way, the GRU outputs, h_t, at each time step of each interaction can be compared to the corresponding ground truth memory labels at the same time step of the same interaction.

The GRU units with the maximum mutual information for each of the 10 memory labels are show in Table 2.5.

2.7.2 Interpretation of binary GRU gate values

To explore the activity of the 'memory units', the output values of the GRU update and reset gates (z_t and r_t, respectively) were analyzed in addition to the output values of the GRU units themselves (h_t).

The output values of the update and reset gates were in the range (0, 1), since they are determined by a sigmoid function. Moreover, the sigmoid function tends to push

the gate outputs to either 0 or 1. Indeed, the histograms of the update and reset gate output values for all time steps for all interactions of the training data (Fig. 2.4) show that the values are usually close to either 0 or 1. Only 1.5% of the update gate outputs and 0.6% of the reset gate outputs did not fall into either the first or the last bin of the histogram. This allows us to treat the gate outputs as if they were binary.

Treating the gate output values as if they were binary greatly simplifies the interpretation of the GRU's activity. With two gates and binary values, there are four possible states that the gates can be in. Actually, two of the states have the same function, so the gates can be thought of as being in one of three possible states at each time step of an interaction, as shown in Table 2.6. The three possible states of the gates are *preserve* memory, *augment* memory, and *overwrite* memory. The functions of these states are shown in Table 2.6.

This GRU analysis method could be applied to any problem where the gate outputs tend to be binary (i.e. close to 0 or 1). Theoretically, the gate outputs can be anywhere in the range (0, 1) (i.e. non-binary). However, for the purpose of interpretability, training techniques for pushing the gate outputs to either 0 or 1 have been proposed [76]. Therefore, our GRU analysis method could be applied to a wide variety of problems by using such training techniques.

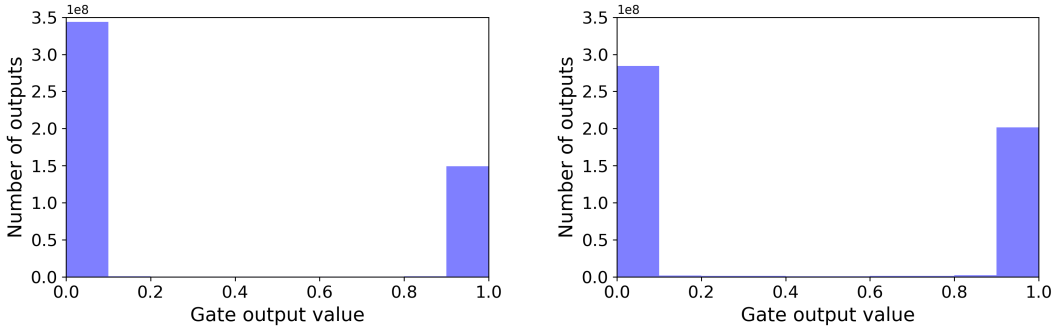


Figure 2.4: Histograms of the GRU's update (left) and reset (right) gate output values from the training data show that the gate values almost always fall into the first or last bin. That is, they are usually close to 0 or 1. Therefore, the gate values can be treated as binary values for this data.

2.7.3 Memory visualization

With the binary gate interpretations, it is possible to visualize the GRU output activations, h_t, over the course of the interaction in a legible way. Two visualizations are shown in Fig. 2.5.

The two grids in Fig. 2.5 display data from two of the simulated training interactions. Short interactions of the same length were chosen to easily compare the behaviors of the memory units for two different interactions.

Update value	Reset value	Function in GRU	Interpretation
0 0	1 0	Previous state h_{t-1} becomes new state h_t (no state change).	*Preserve* memory
1	1	Current input x_t and previous state h_{t-1} are combined to create new state h_t.	*Augment* memory
1	0	Previous state h_{t-1} is discarded and new state h_{t-1} is computed from current input x_t.	*Overwrite* memory

Table 2.6: Interpretation of the binary GRU gate outputs. The leftmost column contains the binary values of a GRU's *update* (z_t) and *reset* gates (r_t), the middle column describes the function of the gates when they have certain values, and the last column contains how the gate functions are interpreted when considering *memory* in interaction.

In the top interaction, the customer states that they are looking for a camera suitable for an expert user (c_t at t=2). The network recognizes the occurrence of a memory-setting action based on the words in the customer's statement (e.g. "looking for", and "expert"). Later in the interaction, the proposed system (shopkeeper), having remembered the customer's preference, is able to introduce some information about the Canon camera relating to expert users (predicted s_t at t=3). Similarly, in the bottom interaction, the customer states that they are looking for a camera for photographing sports (c_t at t=2) and later the proposed system, having remembered, is able to refer to the customer's preference when introducing the autofocus feature (which helps take pictures of moving subjects) (predicted s_t at t=3). In both these examples, the proposed system learned to remember the memory-setting actions and later output appropriate memory-dependent actions.

2.7.3.1 Description of the visualization

The grids in Fig. 2.5 show the activity of the memory units during the interactions. Each row in the grids represents a time step in the interaction, starting from the top and ending at the bottom. The vertical axis labels show the shopkeeper and customer's behaviors that are input to the neural network at each time step, s_{t-1} and c_t. Additionally, the network's shopkeeper action predictions, s_t, are also shown. Note that sometimes the predicted s_t does not match the input s_{t-1} in the next time step because the offline action predictions do not always match the simulated shopkeeper actions, which were used as input to the neural network.

Each column of Fig. 2.5 represents a memory unit from Table 2.5. Each grid cell displays the state of the unit's gates with colored arrows – white (*preserve* memory), gray (*augment* memory), or black (*overwrite* memory). The remaining part of the cell represents the unit's output activation h_t – either -1 (blue) or 1 (red). Since a GRU's output activation is determined by an hyperbolic tangent function (tanh), the activations tend to be pushed to one of the extremes of its range.

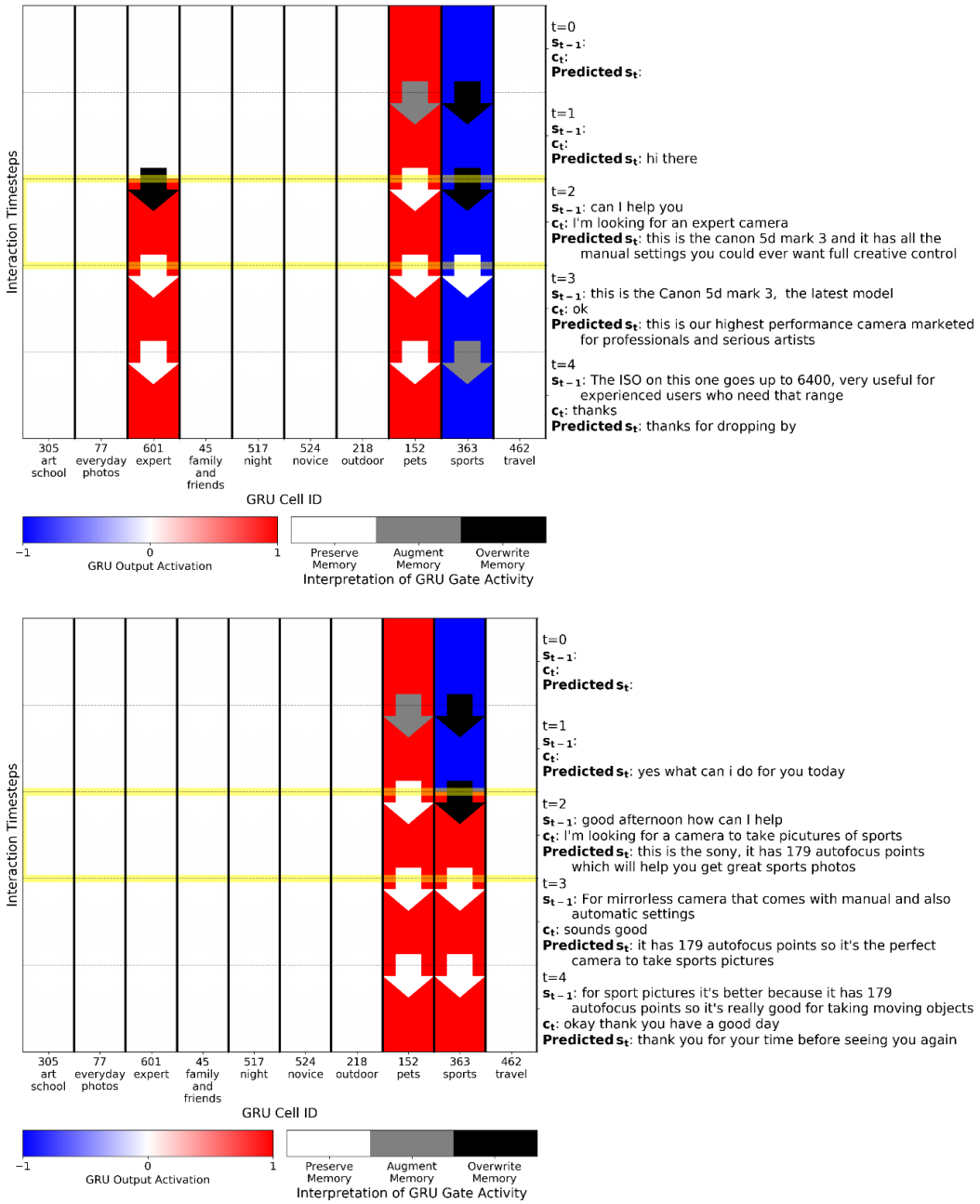


Figure 2.5: Visualizations of the GRU units most important for encoding memories in two interactions. s_{t-1} and c_t are the previous shopkeeper action and current customer action (inputs to the neural network). Predicted s_t is the predicted next robot action (output from the neural network). The time steps with memory-setting actions are highlighted in yellow. Each grid cell represents the internal state of a memory encoding GRU unit at the specified time step. The arrows represent the state the GRU unit's gates and the cell color represents the output activation (h_t), per the color-bar keys.

2.7.3.2 Remembering a memory-setting action

The rows at t=2 are highlighted in yellow to indicate the occurrence of memory-setting actions. In the top grid, at this time step the customer states, "I'm looking for an expert camera". At this point, the GRU layer should encode a memory of 'expert user' so that the robot can subsequently perform appropriate memory-dependent actions. Note the activity of GRU unit 601 'expert user'. At t=2 the gates of unit 601 'expert user' are in the *overwrite memory* state, meaning that the previous GRU state, $h_{601,t=1} = 0$, is overwritten with information from the current input to the GRU layer, $x_{601,t=2}$. This causes the current GRU output, $h_{601,t=2}$, to change to 1. Since the GRU layer input, $x_{601,t=2}$, contains a condensed representation of the current network input, s_1, c_2, it can be inferred that $h_{601,t} = 1$ indicates a memory of the occurrence of these input actions. The bottom grid shows a similar interaction, but where the relevant memory is that the customer wishes to take pictures of 'sports'.

The default activation of a memory encoding GRU unit may be 0, 1, or -1. In the majority of the GRU units visualized in Fig. 2.5 the default activations are 0 (represented by white grid cells), but for GRU units 152 'pets' and 363 'sports' the default activations are 1 (red) and -1 (blue), respectively. Note that the activation for the 'sports' unit in the second interaction changes to 1 when the corresponding memory-setting action occurs. Since there is no occurrence of a 'pets' memory-setting action, the 'pets' unit activation remains 1 throughout the interactions. Since the weights connecting the GRU layer's outputs to the final, output layer of the network, which decides the robot's next action, may be either positive or negative, there is no constraint on the values with which a GRU encodes memory.

2.7.3.3 Inferring the meaning of a memory unit

The network inputs that have been stored in memory during any time step can be identified from a visualization like Fig. 2.5. To determine the contents of a memory unit at time t, trace back through the memory unit's previous activations until the most recent memory *overwrite*, taking note of any memory *augmentations* on the way. The activation of the memory unit at time t represents the occurrence of customer or shopkeeper actions input at the times of the *overwrite* and *augmentations*. In this way, it may be possible in the future to display a GRU's memories in a readily human-readable way.

2.7.4 Distributed representation of memory

Neural networks generally represent information in a distributed manner. The visualizations of GRU activations during the two interactions in Fig. 2.5 demonstrate an ideal case, where a single GRU is sufficient to represent the occurrence of a memory-setting action (e.g. "I'm looking for a camera to take pictures of *sports*."). Further analysis reveals that the memories of such occurrences are distributed over many GRUs.

Fig. 2.6 shows the mutual information (MI) between each memory type and each GRU unit. The average MI was 0.016 bits (s.d. 0.009), the maximum was 0.116 bits,

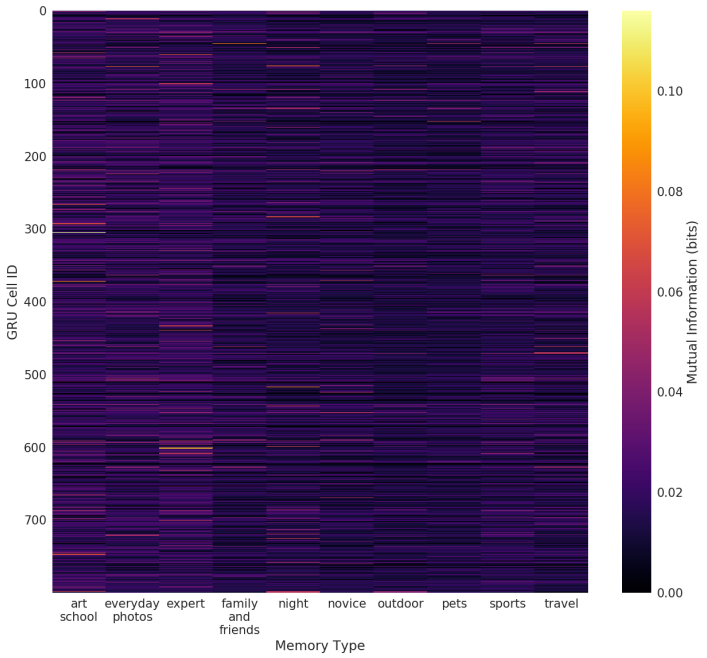


Figure 2.6: Heat map showing mutual information (MI) between memory labels and GRUs units. Units with high MI are important for representing memories. Memories have a distributed representation.

and the minimum was 0.0. This means that on average, 63.5 GRUs would be required to encode one memory type (since ideally a single bit could be used to encode whether or not the memory-setting action occurred). In fact, the distribution of MI over the GRUs is non-uniform, with most GRUs having very low MI and very few with high MI. This distribution can be seen more clearly in Fig. 2.7.

2.8 Discussion

2.8.1 The effectiveness of the proposed system in learning a memory model

This study found that a robot was successfully able to learn a memory model from a large dataset of simulated example interactions that enabled it to correctly perform actions based on memories.

From the analysis in Sec. 2.7 it can be concluded that memories were encoded in the GRU layer of the robot's neural network. That is, some GRU units encoded long term memories about the customer's actions. These GRU units work like a binary switch: when 'on' the unit represents that a specific memory-setting action occurred (e.g. the

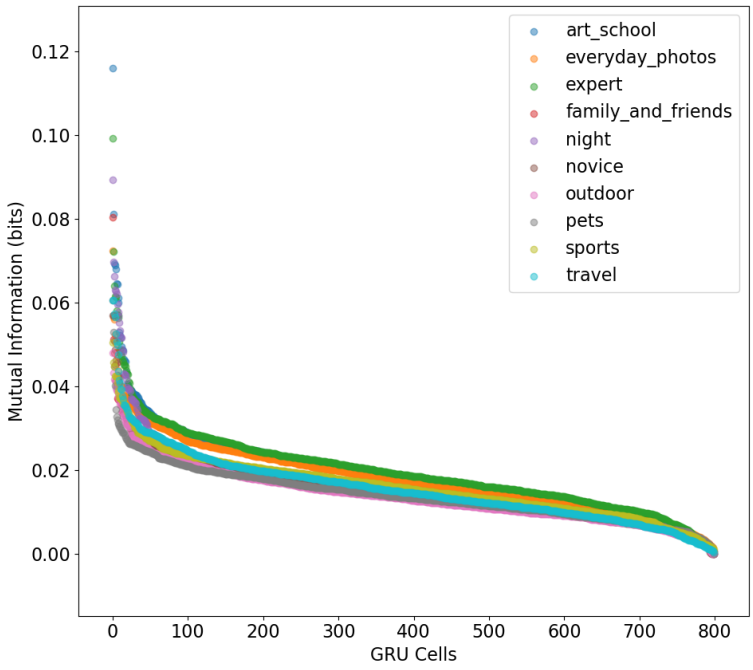


Figure 2.7: Scatter plot showing mutual information (MI) between each GRU unit and each memory label. Points have been sorted by MI to better show the distribution of MI over the GRUs. Only a few GRUs have high MI and most have low MI, showing that some GRUs are specialized for representing the occurrence of memory-setting actions.

customer stated a preference for cameras well-suited for an expert photographer), and when 'off' it represents that the specific memory-setting action had not occurred. The final output layer of the neural network, which has connections leading from the GRU layer and decides the robot's next action, uses the information stored in the GRU units to determine whether it is appropriate to perform a memory-dependent action, and which memory-dependent action is the most appropriate. The results of the offline evaluation, which showed that the system with a GRU layer for encoding memories significantly outperformed a system without a GRU layer, supports this conclusion.

In an ideal case, the number of GRU units that learn to encode memories should match the number of memory types that occurred in the training dataset. However, since the utterances in memory-setting actions have many natural variations of phrasing, it is challenging for the neural network to learn a perfect memory representation. For example, in the analysis in Sec. 2.7 only the units that were most critical for encoding the memories (based on mutual information with the ground truth memory labels) were examined, but the representation of memory was also distributed over some other GRU units with relatively high mutual information, as can be seen in Figs. 2.6 and 2.7.

This sort of distributed representation, which is common in neural networks, provides a challenge for future work to visualize memories during human-robot interaction at greater levels of detail.

2.8.2 Generalizability and scalability

In the future, it would be interesting to apply the proposed system to other domains and real-world scenarios. In this work the proposed system was trained and tested on simulated interactions in a simplified scenario, and we believe it could successfully be applied to real-world scenarios. However, there are some limitations.

The proposed system was able to model several different memory types, which we expect would also occur in real-world camera shop interactions (Table 2.1). For example, we identified actions that depend on memories of the customer's level of knowledge, types of pictures the customer prefers to take, and camera features that the customer wants. In a general, real-world shopping scenario, we expect that the types of memories would be similar to what is found here; although, the number of customer types could be greater. This would necessitate a larger quantity of training data.

It is possible to estimate the required quantity of interaction data to train the proposed system. If there are m memory types, c customer types, s natural phrasings of each memory-setting action, d natural phrasings of each memory-dependent action, an average of i memory-dependent actions per interaction, and k examples are required by the machine learning algorithm to learn an accurate representation, then roughly $\frac{kmcsd}{i}$ training interactions would be required. In this work, we simulated interactions to get a sufficient number of training examples. To train the system for real-world scenarios that contain a greater variety of customer types and phrasings, a larger quantity of training data would be required. We believe that it will be possible to collect such datasets in the future with passive sensors set up in areas with high customer traffic.

One limitation of the proposed system is that it cannot generalize to memory types that are underrepresented in the training interactions. For example, the system cannot customize its behavior to idiosyncratic, infrequent customer types, such as *a customer seeking a gift for their father to use while scuba diving*. This is a limitation of data-driven HRI in general, since systems such as these are designed to learn only the repeatable, core behaviors of interaction. In the future it would be useful to detect idiosyncratic customer behaviors so the robot can take appropriate action. Furthermore, it may be necessary to combine the data-driven approach with other approaches, such as knowledge-based approaches, to deal with idiosyncratic situations.

2.9 Conclusion

In conclusion, this work attempts to answer the question of whether or not a shopkeeper robot is able to learn a memory model of human customer behavior to enable it to appropriately perform actions that depend on memory. To get a foothold in answering

this question, a dataset of human-human interactions in a camera shop scenario was analyzed to discover the types of actions that involve memory. The analysis revealed several types of *memory-setting* and *memory-dependent* actions. Due to the wide diversity of natural utterance variations that occurred in the memory-setting and memory-dependent actions in the human-human dataset, and the variety of possible interaction patterns, the number of examples of each interaction pattern was deemed too few to train a machine learning algorithm. Therefore, a dataset of simulated interactions was generated for that purpose. To train a shopkeeper robot, a learning system with a gated recurrent unit (GRU) recurrent neural network was proposed. The proposed system was compared to a baseline system without GRUs in an offline evaluation. A human evaluator found the actions of the proposed system to be correct at a significantly higher rate than the actions of the baseline. The trained neural network was analyzed with a visualization to reveal how it represented memories of customer behavior in its GRU layer. Finally, the results were discussed and some possibilities for future work were suggested.

2.10 Appendix – Tables used for interaction simulation

Action	Actor
ENTERS	Customer
GREETS	Customer, Shopkeeper
WALKS_TO_CAMERA	Customer
LOOKING_FOR_A_CAMERA	Customer
LOOKING_FOR_A_CAMERA_WITH_X	Customer
EXAMINES_CAMERA	Customer
JUST_BROWSING	Customer
LEAVES	Customer
QUESTION_ABOUT_FEATURE	Customer
ASKS_FOR_SOMETHING_WITH_MORE_X	Customer
SILENT_OR_BACKCHANNEL	Customer
THANK_YOU	Customer, Shopkeeper
THINK_IT_OVER	Customer
DECIDE_TO_BUY	Customer
ANSWERS_QUESTION_X	Customer
NONE	Shopkeeper
ASKS_QUESTION_X	Shopkeeper
RETURNS_TO_COUNTER	Shopkeeper
LET_ME_KNOW_IF _YOU_NEED_HELP	Shopkeeper
ANYTHING_ELSE_I_CAN_HELP_WITH	Shopkeeper
ANSWERS_QUESTION_ABOUT_FEATURE	Shopkeeper
INTRODUCES_NEW_FEATURE	Shopkeeper
INTRODUCES_CAMERA	Shopkeeper
THIS_IS_MOST_X	Shopkeeper
APPENDED_PHRASE_RELATING_TO_MEMORY	Shopkeeper

Table 2.7: Simulator actions

Features	Feature is preferred for picture type									
	Art school	Everyday photos	Expert user	Family and friends	Nighttime	Novice user	Outdoors	Pets	Sports	Travel
Point and shoot	No	Yes	No	Yes	No	Yes	No	No	Yes	Yes
Mirrorless	No	No	No	No	No	No	No	No	No	No
DSLR	No	No	No	No	No	No	No	No	No	No
Colorful	No	No	No	No	No	No	No	No	No	No
Black	No	No	No	No	No	No	No	No	No	No
White	No	No	No	No	No	No	No	No	No	No
Silver	No	No	No	No	No	No	No	No	No	No
Purple	No	No	No	No	No	No	No	No	No	No
Pink	No	No	No	No	No	No	No	No	No	No
Red	No	No	No	No	No	No	No	No	No	No
Light	No	No	No	No	No	No	Yes	No	No	Yes
Heavy	No	No	No	No	No	No	No	No	No	No
Preset modes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	No	Yes
Glamour retouch effects	No	Yes	No	Yes	No	Yes	No	Yes	No	No
Artistic modes	No	Yes	No	Yes	No	Yes	No	Yes	No	No
Cheap	Yes	No	No	No	No	Yes	No	No	No	No
Expensive	No	No	No	No	No	No	No	No	No	No
Autofocus points	No	Yes	No	Yes	No	No	No	Yes	Yes	No
Long exposure	Yes	No	Yes	No	Yes	No	Yes	No	No	No
Full manual control	Yes	No	Yes	No	No	No	Yes	No	Yes	No
Full frame	Yes	No	Yes	No	No	No	Yes	No	No	No
High ISO	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No
Bulb mode	No	No	Yes	No	Yes	No	No	No	No	No

Table 2.8: Picture types and preferred features

Features	Camera has feature		
	Nikon	Sony	Canon
Point and shoot	Yes	No	No
Mirrorless	No	Yes	No
DSLR	No	No	Yes
Colorful	Yes	No	No
Black	Yes	Yes	Yes
White	No	Yes	No
Silver	Yes	Yes	No
Purple	Yes	No	No
Pink	Yes	No	No
Red	Yes	No	No
Light	Yes	Yes	No
Heavy	No	No	Yes
Preset modes	Yes	Yes	No
Glamour retouch effects	Yes	No	No
Artistic modes	No	Yes	No
Cheap	Yes	Yes	No
Expensive	No	No	Yes
Autofocus points	Yes	Yes	No
Long exposure	No	Yes	Yes
Full manual control	No	Yes	Yes
Full frame	No	No	Yes
High ISO	No	No	Yes
Bulb mode	No	No	Yes

Table 2.9: Camera features

Chapter 3

Curiosity-Based Learning with a Time-Persistent Customer Behavior Model

Abstract – Learning from human interaction data is a promising approach for developing robot interaction logic, but behaviors learned only from offline data simply represent the most frequent interaction patterns in the training data, without any adaptation for individual differences. We developed a robot that incorporates both data-driven and interactive learning. Our robot first learns high-level dialog and spatial behavior patterns from offline examples of human-human interaction. Then, during live interactions, it chooses among appropriate actions according to its curiosity about the customer's expected behavior, continually updating its predictive model to learn and adapt to each individual. In a user study, we found that participants thought the curious robot was significantly more humanlike with respect to repetitiveness and diversity of behavior, more interesting, and better overall in comparison to a non-curious robot.

Keywords: Data-driven social interaction, curiosity-based learning

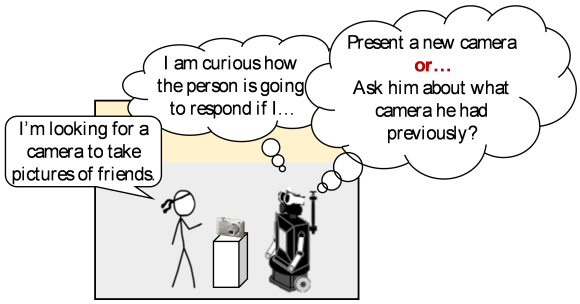


Figure 3.1: A curious robot chooses among the appropriate actions to satisfy its curiosity about the customer's individual differences during interaction.

3.1 Introduction

In recent years, opportunities for the public to interact with social robots have become more common, with robots appearing in education [15, 56], entertainment [63, 87], and in the service industry [114, 127]. Thanks to the increasing availability of big data, one promising approach to train conversational robots to interact autonomously with humans is to automatically learn replicable dialog patterns (e.g. question-answer) from observations of real human-human interaction, as demonstrated in prior work for robots in the role of a shopkeeper [78], an assistant [9], a bartender [96], and a storyteller [75].

While existing data-driven approaches enable a robot to generate socially appropriate behaviors, such learning-based techniques only learn from previously collected data [1, 9, 75], but are not able to continuously adapt to human actions during live interaction once the initial interaction strategies have been learned. This can result in a robot that behaves the same way regardless of the human's behaviors, which sometimes yields monotonous, boring interactions. To illustrate, imagine the following conversation (Fig. 3.1) between a customer and a shopkeeper in a camera shop:

Customer: I am looking for a camera to take pictures of friends.

Shopkeeper: What sort of camera did you have before?

Customer: I just used my phone to take pictures.

. . . after customer and shopkeeper talk for a while. . .

Customer: So anyway, I'm looking for a camera to take pictures of friends.

In this example the shopkeeper starts his line of questioning believing it will lead to a sale. However, when the sale does not occur, and the customer later restates his purpose, the interaction becomes a learning opportunity for the shopkeeper. At this point, the human shopkeeper, having already tried the most promising strategy to make a sale, will be driven by his *intrinsic motivation* to try different actions, which could possibly lead to a favorable outcome. From a learning perspective, this can be seen as an exploration of the interaction space, probing to see how the customer will react to various actions, such as presenting the customer with a camera or leaving the customer to look around on their own. In machine learning, the concept of 'curiosity' has been defined to represent an intrinsic reward signal which motivates an agent to explore a feature space in such a way [57, 94, 110].

Inspired by our own human sense of curiosity, we propose an approach that enables a robot to learn continuously through interaction, driven by an entropy-based 'curiosity' mechanism. Nonetheless, 'curiosity' should not *kill* the robot. That is, the robot should not randomly or naively explore behaviors in an unconstrained space such that it fails its job as a shopkeeper (e.g. saying "goodbye" when the customer enters the shop). To ensure that the robot's responses remain task-appropriate, the structure of common dialog patterns for reproducing high-level behavior can be learned *a priori* [78, 79]. Then, during interaction, the robot can learn about the customer's individual differences, driven to explore different behaviors by the 'curiosity' mechanism while being constrained by the appropriate pre-learned patterns. Our goal in this work is not to fully replicate human curiosity, not to exploit learned information in goal-driven behavior, nor to pursue

specific customer information. Rather, our aim is to emulate curiosity sufficiently to drive the robot to explore a variety of behaviors and learn about customers' individual differences, in order to yield more human-like, interesting human-robot interactions.

3.2 Related Work

3.2.1 Data-driven Approaches for Conversational Agents and Social Robots

There exist some dialog systems that learn the state of a conversation and model it over time, so that they can always perform the most appropriate action; however, the states, which are formulated as slots and values [134], depend on human designers to specify domain-specific knowledge and annotate training data, so they are not suitable for unsupervised learning of robot behaviors. In other cases, the systems have not been evaluated in live interactions with humans [10, 53]. For social robots, several studies have aimed to learn robot behaviors for completing the same task as a human from data collected from online games [9, 17, 125], remote web users [75], and real human interaction data [1, 29, 91]. Thomaz et al. developed a framework for online users to provide feedback to a Reinforcement Learning (RL) agent learning to perform tasks observed in a game [123]. Leite et al. proposed a semi-situated learning method to crowdsource dialog lines from multiple authors, with each dialog line associated with a goal-directed action [75]. Our work complements these approaches by considering data collected directly from human-human interaction in a physical environment; however, we are also interested in a robot that continues to learn based on its intrinsic motivation, to better adapt to each user over time.

3.2.2 Curiosity-driven Learning

There is some work, especially in the field of sensorimotor learning, on generating autonomous agent behaviors based on intrinsic motivation, without pre-programmed goals [57, 94, 110]. Oudeyer et al. developed a RL algorithm with the objective of maximizing learning progress [94]. Kaplan and Oudeyer demonstrated that a robot is able to explore within its repertoire of motor primitives [57]. Action selection strategies based on information gain have also been applied for spatiotemporal exploration by mobile robots in continuously changing environments [89, 108]. The interactive art sculpture presented in is a distributed system with a large sensorimotor space that employed a similar concept of curiosity-based learning. Their experiment demonstrated that the sculpture gradually shifted to more exploratory actuation patterns as it learned about its own mechanisms and surroundings through self-experimentation and interaction.

Hester et al. presented an RL model that used intrinsic rewards to explore uncertain sensorimotor spaces and acquire new experiences for training the model [46]. It was

tested on a robot that learned to hit a cymbal. Qureshi et al. introduced an intrinsic motivation system for a social robot to learn when to perform gestures and facial expressions in real-world interactions based on minimization of state prediction error via RL [99]. Our work also employs the intrinsic drive of curiosity, but for a social robot that generates verbal expressions and nonverbal motion.

Madani et al. proposed a two-level cognitive system for learning language-to-perception groundings with adaptive visual perception cast as "perceptual curiosity" and an evolutionary system to learn grounding functions to guide "epistemic curiosity" [85]. As the system learned, epistemic curiosity drove the robot to ask about objects with low-confidence groundings. In contrast to their approach, our work focuses on social behavior and uses and entropy-based metric to explore low-confidence social spaces.

3.2.3 Robots that Elicit Curiosity in Humans

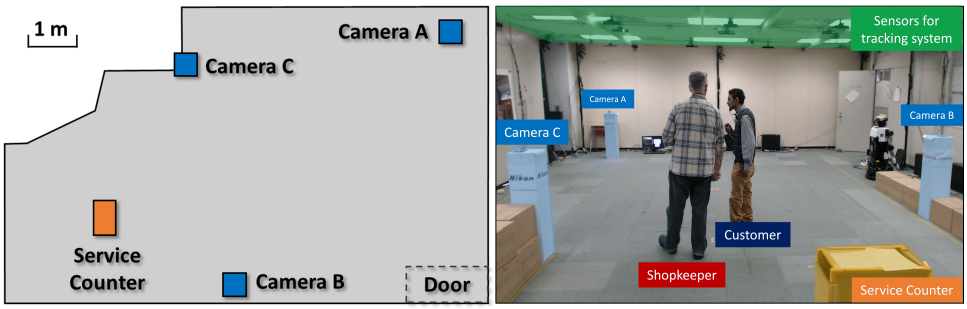
Some studies have focused on eliciting curiosity in humans for the sake of incentivizing productivity or improving social interaction. For instance, eliciting curiosity in humans was investigated for a robot that generated unpredictable responses [72] and for a robot that demonstrated curiosity in its verbal expressions [36]. Their results showed that such robot behaviors positively influenced the user's experience during interaction. Likewise, we expect that a curiosity-driven robot will have a similar effect. The originality of our work lies in generating curiosity-driven behaviors via continuous learning, rather than manually implementing pre-scripted verbal expressions.

3.2.4 Active Learning for Social Robots

There is work in the area of active learning [113] on how robots can increase their overall knowledge based on asking appropriate questions, choosing what questions to ask, and choosing when to ask things. For example, Cakmak and Thomaz identified the types of questions that a robot could ask to guide active learning of new skills, explored their use in human communication, and evaluated human perceptions of robots that ask such questions [13]. Thomason et al. has shown that an 'inquisitive' robot that uses active learning to learn language-grounding functions is judged to be more fun to interact with when it asks off-topic questions [122]. The scenarios and goals in these works are different from our work, but there is overlap in choosing among robot actions in order to efficiently explore a space and improve a model based on interaction with humans.

3.3 Human-human Interaction Dataset

A dataset of human-human interactions, described below, was used for the purpose of training the data-driven robot behavior system, to be presented in Sec. 3.4. We used the dataset originally presented in [82].



(Customer enters the shop and shopkeeper approaches)
C: I'm looking for a camera that is stylish, affordable, and easy to use. Do you have one of those?
S: Depends on what sort of pictures you take, we have two very stylish affordable cameras, one is $68 and one is over $500.
C: Oh I see, I think I might be looking to get the one that costs 68 dollars, can you please show me?
S: (Present Nikon) Sure, just over here this camera takes beautiful pictures but it's designed to be point-and-shoot, catch the moment.
C: Can you explain to me about the preset modes?
S: It has 18 different modes where you can tell it what sort of conditions you are in ...
(Customer asks a few more questions about camera features and shopkeeper answers)

Figure 3.2: An example interaction observed in our data collection

3.3.1 Scenario

To observe typical interaction patterns, a camera shop environment was setup in an 8m x 11m experiment space with three camera models, each at a different location. For each interaction, one shopkeeper participant interacted with one customer participant. An interaction example is shown in Fig. 3.2.

3.3.2 Sensors

The participants' speech and position data were recorded as they interacted with each other. A sensor network with a human position tracking system recorded the participants' location data [11]. To capture the participants' speech, they were asked to speak into handheld smartphones running the Google speech recognition API. To detect the start and stop of speech activity, participants were required to touch the mobile screen to indicate the beginning and end of their speech.

3.3.3 Participants

Fluent English speakers were recruited as participants for the role of the customer. They had varied levels of knowledge about cameras. A total of 18 participants were employed (13 male, 5 female, average age 32.8, s.d. 12.4). To obtain a diverse set of behaviors, two different participants were chosen to role-play as the shopkeeper. They had very different interaction styles, one with a more outgoing personality (male, age 54) and another with a quieter disposition (female, age 25).

3.3.4 Procedure

The participants were encouraged to act naturally and focus discussion on the features listed on the camera spec. sheets (8 to 10 features per camera). Customer participants were encouraged to play with the cameras, browse the shop, or ask camera-related questions. They role-played in different interactions as advanced or novice camera users in order to keep them interested and minimize fatigue. The shopkeeper participants were instructed to wait at the service counter at the start of the interaction, be polite, and behave according to their role (e.g. greetings and farewells, letting the customer browse, answering questions, or introducing products when appropriate). As the example in Fig. 3.2 shows, participants used a variety of fillers (e.g. "you know", "like") and backchannels (e.g. "I see") in their utterances. Therefore, we believe our setup elicited reasonably natural behaviors.

Each shopkeeper interacted with 9 different customer participants. Each customer role-played 24 interactions (12 as advanced and 12 as novice) for a total of 432 interactions. 27 interactions were removed (16 due to technical failures and 11 due to customers that did not follow instructions). In total, we collected 405 interactions, with 4061 shopkeeper utterances and 4115 customer utterances[11].

3.4 Proposed Technique

In order to develop a curious robot that continues to learn during interaction, we used a series of techniques that enable both behaviors and interaction logic to be directly learned from noisy sensor data without human intervention. The system architecture is shown in Fig. 3.3.

First, using data abstraction techniques from previous work [78], the raw data was processed into a form suitable for input and output to two neural networks.

Then, although the behaviors of the robot are to be driven by curiosity, the robot should still follow the social rules observable in the human-human data. So, we trained an *Appropriateness Learner* (the first neural network) on the processed human-human interaction data, for the purpose of constraining the robot's actions to a subset of socially appropriate actions that it can curiously explore in a particular situation.

Next, we developed a *Curiosity Learner* (the second neural network) to assign a curiosity score to each possible robot action. We model curiosity as the drive to minimize the variance of the prediction error of the consequence of the robot's actions. Similar to the shopkeeper, a customer will generally behave within the norm of certain interaction patterns, which can be used as a prior for the *Curiosity Learner*. However, due to individual differences, different customers may react very differently to a given robot action, and it is these differences to which a curious robot must tailor its interaction dynamics. Thus, during live interaction, the *Curiosity Learner* is updated.

[11] The dataset can be obtained here: http://www.geminoid.jp/dataset/camerashop/dataset-camerashop.htm

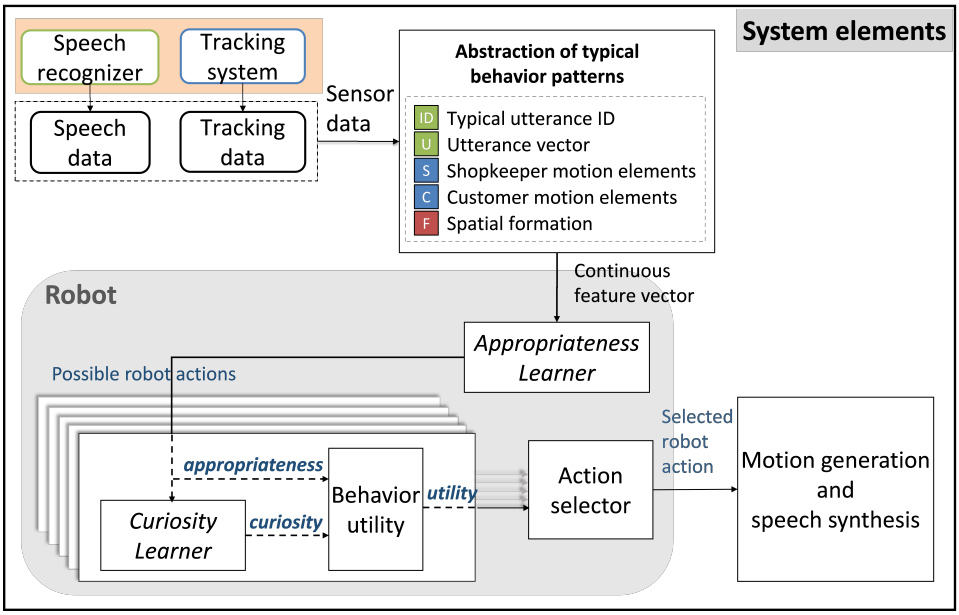


Figure 3.3: Architecture of the proposed system elements

3.4.1 Data Abstraction and Representation

Behavior abstraction was applied to the raw interaction data in order to reduce the data's dimensionality and the effects of sensor noise, thus simplifying the learning problem. Behavior abstraction was accomplished by action segmentation, motion target and stopping location clustering, applying models of spatial formations, and action clustering, as originally presented in previous work [78]. Here we describe how the data abstraction techniques were used to prepare the input vectors and training targets for the Appropriateness Learner and Curiosity Learner neural networks. An example of our abstracted representation is shown in Fig. 3.4.

3.4.1.1 Motion and Spatial Formation Abstraction

The participants' common motion targets and stopping locations in the camera shop environment were discovered with unsupervised clustering. Spatial formations *presenting object*, *face-to-face*, and *waiting* were computed using pre-existing HRI models [135], [42], and [61], respectively. Below we refer to the formations as *spatial states*. Furthermore, a *state target* is specified for the *presenting object* formation, i.e. the object being presented.

3.4.1.2 Action Sequence Discretization

Each interaction was discretized into a sequence of alternating human and robot actions, with an action identified whenever a participant: 1) speaks an utterance, 2) changes their motion target, or 3) yields their turn by allowing a period of time to

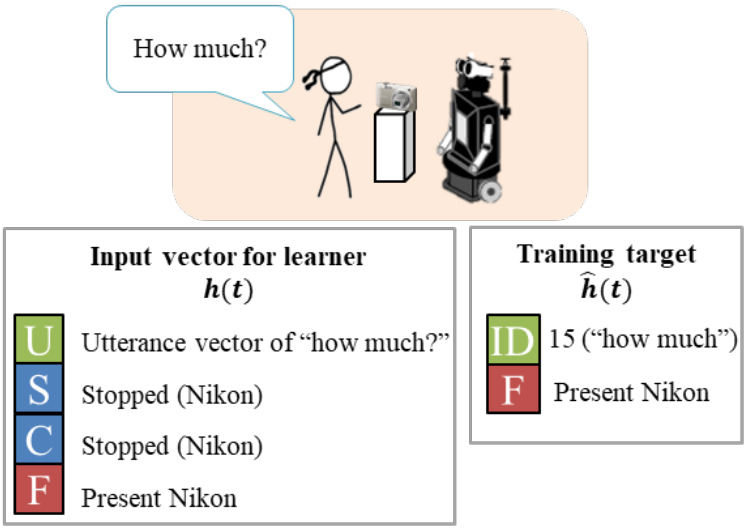


Figure 3.4: Example of input vector and training target. Here, "S" and "C" denote shopkeeper and customer motion elements, "F" denotes spatial formation, and "ID" denotes typical utterance ID.

elapse with no action, which we define as a yield action. We denote an interaction as $(h(t), r(t), h(t+1)\ldots)$, where $h(t)$ represents the human action and $r(t)$ represents the robot action at time t. Each action consists of the participant's speech, current location, movement arrival location, movement departure location, spatial state, and state target (in the case of *presenting object*).

3.4.1.3 Action Vectorization

Each action was represented as a vector for input to the *learners* and for action clustering. The vectorization procedure is the same as in [78, 79].

The human customer and shopkeeper utterances were vectorized using common text-processing techniques. First, stop words were removed from the utterances, a stemmer was applied, and n-grams were computed for n=1, 2, and 3. Next, latent semantic analysis (LSA) [68] was used to reduce the n-gram vectors to 1000 dimensions. To emphasize important words, keywords were extracted from the utterances using a cloud-based API (now part of IBM Watson[2]) to create a separate keyword vector, which was then reduced to 200 dimensions using LSA. In this way, each utterance was represented with a 1200 dimension vector. Customer and shopkeeper utterance vectorization were performed separately.

The shopkeeper and customer spatial information was vectorized into a vector containing the participant's location (7 dims.), movement departure location (7 dims.), and movement arrival location (7 dims.). Furthermore, the shopkeeper and customer's joint

[2] https://www.ibm.com/watson/services/natural-language-understanding/

spatial state, consisting of spatial formation (3 dims.) and state target (4 dims.), was also included. Thus, customer actions and shopkeeper actions were both 1228 (m) dimensions.

3.4.1.4 Action Clustering

Each action was represented using a discrete ID, obtained by action clustering, for the output of the *learners*. The goal of action clustering was to group together similar actions from the human-human dataset to find sets of discrete customer and shopkeeper actions that commonly occurred during the interactions. The shopkeeper action cluster IDs were used as outputs for the Appropriateness Learner and the customer action cluster IDs were used as outputs for the Curiosity Learner.

Actions were clustered into clusters of similar actions by BIRCH clustering [137]. The number of clusters, K, was set to 800. The branching factor was set to 10 and the threshold was 0.005.

Since the system's output, a shopkeeper action cluster ID, must dictate the robot's behavior, which includes speech, destination location, and spatial formation, a *typical action* was automatically selected for each shopkeeper action cluster by finding the action that was most similar to all the other actions in the cluster (i.e. the medoid) using cosine distance. Since complete utterances with few ASR errors tended to share the most similarities with other utterances in the same cluster, the utterances of typical actions tended to be well formed and easy to understand. We refer to these as *typical utterances*. When the system outputs a shopkeeper action cluster, the robot speaks the *typical utterance* and moves based on the spatial portion of the *typical action*.

The number of clusters, K, was set to be 800. This value was chosen to be larger than in previous work [78, 79] in order to obtain more fine-grained action clusters that account for more variability in speech. For example, a large number of shopkeeper clusters provides the shopkeeper with more possible subtle variations of behavior, such as saying, "This camera is $500" versus, "Let me tell you about the price of this camera, it's $500." Moreover, a large number of customer clusters helps account for individual differences in phrasing, e.g. "What is the price?" versus "Tell me the price."

3.4.2 Learning from Example Interactions

Here, we describe the details of the individual components of the curiosity-based learning system, which is illustrated in Fig. 3.5.

3.4.2.1 Appropriateness Learner

First, to learn the social appropriateness of a robot action, we applied a feed-forward multilayer perceptron neural network, which has the ability to learn a mapping, based on the relative importance of each input feature, to a discrete class, from examples in a dataset $\mathscr{D}$. Our training data for the neural network is composed of $(h(t),\hat{r}(t))$ action pairs, where $h(t) \in \mathbb{R}^m$ is the human action input vector and $\hat{r}(t) \in \{0,1\}^K$ is a target

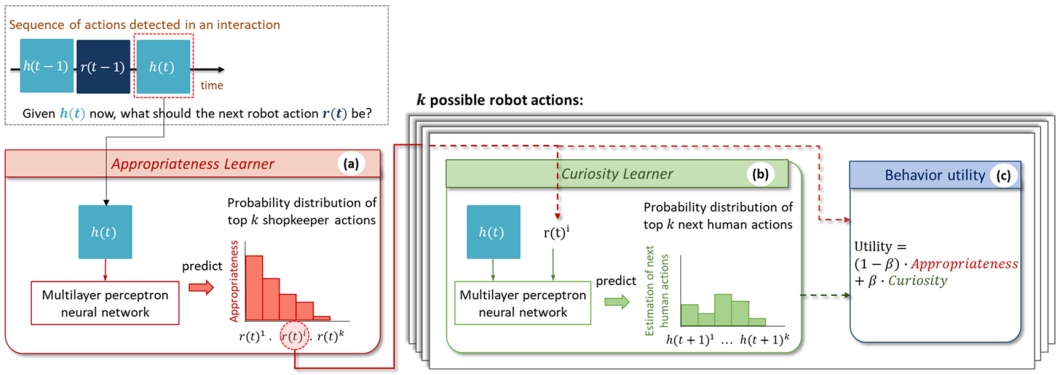


Figure 3.5: Details of our system, both the Appropriateness Learner and Curiosity Learner is triggered when a human action (e.g. utterance or silence) is detected. (a) The Appropriateness Learner learns a set of k socially appropriate robot actions, and for each robot action, (b) we query the Curiosity Learner to output a measure for curiosity (c) finally a utility measure is calculated.

robot shopkeeper action, where K is equal to the total number of robot actions obtained from clustering. That is, if $\hat{r}(t)^i = 1$, observation $h(t)$ maps to robot action i.

Based on the results from previous studies, we can interpret the neural network as learning a measure of how *appropriate* each robot action is :

$$Appropriateness = p\left(r(t)^1\right), \ldots, \; p\left(r(t)^K\right) \tag{3.4.1}$$

During online interaction, we want to constrain the robot to a number of possible behaviors that are socially appropriate for a particular situation. Thus, we only select the top k most appropriate robot actions predicted by the neural network, among which the robot can freely explore using the *Curiosity Learner*.

3.4.2.2 Curiosity Learner

We model the curiosity of a robot action as trying to minimize the variance of the prediction error [14], i.e. the robot is curious about those actions for which it is uncertain how the customer will respond, and less curious about actions for which it is confident it can predict what the customer will do next. Similar to the *Appropriateness Learner*, we can learn an initial estimation of potential next customer actions, by applying a second multilayer perceptron neural network.

Considering a sequence of alternating actions $(h(t), r(t), h(t+1))$, the training input for the neural network is $(h(t), r(t))$, and the training target is the discretized value, $\hat{h}(t+1)$. The neural network learns a probability distribution over the set of human actions in the next timestep, $p(h(t+1)^1)$, $\ldots$, $p(h(t+1)^K)$, where K is the total number of discrete human actions. Finally, only the top k most likely subsequent customer actions were used.

To measure the uncertainty of the customer's next action, we can calculate the entropy of the probability distribution that is output by the neural network [131]. Previous computational models have also incorporated such uncertainty-based strategies, generating biases toward actions or states that have high entropy [21, 104]. A high entropy value means that the robot is unsure what the customer will do as a result of its own action, while a low entropy value means that the robot is fairly confident of what the customer will do next. Within this framework, the robot is encouraged to take actions that result in states that are deemed surprising – i.e., actions for which the robot is unsure what the customer will do next.

Thus, the *curiosity* measure is the normalized entropy of the probability distribution:

$$Curiosity = \frac{-\sum_{i=1}^{k} p'\left(h(t+1)^i\right) \ln(p'(h(t+1)^i)}{\ln(k)} \tag{3.4.2}$$

where $p'(h(t+1)^1)$, ..., $p'(h(t+1)^k)$ is the probability distribution over the top k most likely subsequent customer actions, normalized to sum to 1.

3.4.2.3 Behavior Utility

For each potential robot action, a behavior utility function is evaluated, which combines the factors of *social appropriateness* and *curiosity*:

$$Utility = (1-\beta) \times Appropriateness + \beta \times Curiosity \tag{3.4.3}$$

where β is a tuning parameter that is adjustable. A high β biases the robot to be more curious while a low β biases the robot to be more socially appropriate during interaction.

3.4.2.4 Action Selector

To select a behavior for a robot, the behavior utility function is evaluated for each of the potential actions the robot can perform. The action selector then executes the robot action with the highest utility, which is a discrete action, consisting of a typical utterance and a target spatial formation.

3.4.3 Adaptation to Individuals

As the robot continues to interact with a customer, it should come to better understand how the customer will respond to its actions. Thus, it will tend to be less curious about actions it has taken previously. To reflect this new observation in the *Curiosity Learner*, during live interaction, we can update the weights of the neural network in the *Curiosity Learner* through backpropagation. Backpropagation is used to modify the synaptic weights of the internal (hidden) and output layers of the neural network [101], by trying to minimize the loss between the target and the predicted value. In this way,

Algorithm 1 Robot action selection and online adaptation to customer action

Initialize β, th, l

When new human action $h(t)$ is detected:

// Compute the next robot action

$h(t) \leftarrow vectorize\big(h(t)\big)$ // Vectorize for input to the *Appropriateness Learner*

$r(t)^1, \ldots, r(t)^k \leftarrow AppropriatenessLearner\big(h(t)\big)$ for k robot actions with highest $appropriateness$

for $i \in [1, k]$ **do:** // For each of the most appropriate robot actions

  $r(t)^i \leftarrow vectorize(r(t)^i)$ // Vectorize for input to the *Curiosity Learner*

  $h(t+1)^1, \ldots, h(t+1)^k \leftarrow CuriosityLearner(h(t), r(t)^i)$ for k most likely next human actions

  $curiosity(r(t)^i) \leftarrow \frac{-\sum_{i=1}^{K} p(h(t+1)^i)\log(p(h(t+1)^i)}{ln(K)}$ // where $p(h(t+1)^i)$ is the prob. output by *Curiosity Learner*

  $utility(r(t)^i) \leftarrow (1-\beta) \cdot appropriateness(r(t)^i) + \beta \cdot curiosity(r(t)^i)$

$r(t) \leftarrow \max_{i \in k}(utility\ (r(t)^i))$ // Find the robot action with highest utility and execute

// Do the online learning to update the *Curiosity Learner*

$\hat{h}(t) \leftarrow NearestNeighbor$ of all possible K human actions to $h(t)$ // Find the most similar customer action cluster to $h(t)$

do: // Iteratively train *Curiosity Learner* until stopping condition is met

  Update *Curiosity Learner* by backpropagating with input ($previous\ customer\ input$, $previous\ robot\ input$) and training target $\hat{h}(t)$

  if $training\ loss < th$ or $num.iterations > l$: **break** // Update until training loss thresh. or max. num. iterations

$previous\ customer\ input \leftarrow h(t)$ // update for the next round of online learning

$previous\ robot\ input \leftarrow r(t)$ // "

Figure 3.6: Pseudocode for the curiosity-based learning algorithm for robot action selection and online adaptation to the individual customer

the input-output mapping of the neural network can be dynamically updated to reflect new observations.

To update the weight of the neural network, the newly observed human action is first mapped to an action cluster, $\hat{h}(t)$, using the nearest neighbor algorithm. Then, this action is used as the target for backpropagation, with the previous customer action $h(t-1)$ and robot action $r(t-1)$ as inputs. We used the cross-entropy function to compute the loss. To control how quickly the neural network learns the observed human action, we backpropagate the newly observed interaction data through the neural network over several epochs (to a maximum of l) until the cross-entropy loss is below a certain threshold, th. This allows the recently observed human action to immediately become a much more likely prediction for that prompt.

Because the Curiosity Learner network is trained on only a single training example for a limited number of epochs, the online learning process does not impede the run-time performance. An analysis of the run-time performance is presented below in Sec. 3.5.4.3.

The overall process for the curiosity-based learning system is described in the pseudocode in Fig. 3.6.

3.4.4 Model Parameters

3.4.4.1 For learning from the human-human dataset

To find the ideal parameters for training the neural networks on the human-human interaction dataset, we iteratively tested different parameter values. The parameters were adjusted to improve the Appropriateness Learner's shopkeeper action prediction accuracy and the Curiosity Learner's customer action prediction accuracy.

The architectures of both neural networks in the Appropriateness Learner and the Curiosity Learner are the same, consisting of an input layer, followed by three leaky rectified hidden layers, and a softmax output layer. The input to the Appropriateness Learner is the human action vector of dimension m and the input to Curiosity Learner consists of both a human and a robot action vector with total dimension $2m$. Each hidden layer consists of 800 neurons.

Both neural networks were trained using momentum-based mini-batch stochastic gradient descent, with a batch size of 128, a learning rate of 0.0005, and a momentum coefficient of 0.9. Normalized initialization, described in [51], was used to initialize the neural network. The network was trained to minimize the cross-entropy loss for 2000 epochs between the observed target action and the predicted action for the entire training set.

3.4.4.2 For online learning

The Curiosity Learner's parameters for online learning during interaction were tested in simulation, where a human user was able to choose the actions of the customer and observe the proposed system's responses.

We found 0.05 to be a good threshold, th, and 10 to be a good maximum number of iterations, l, for terminating the iterative backpropagation process. Setting th too high or l too low causes online learning to halt before the Curiosity Learner learns the customer's individual differences. Setting th too low or l too high causes overfitting to the most recently observed customer action, such that the network predicts that customer action regardless of the robot's previous action.

For behavior generation, we tested several values for k, the number of possible robot actions, and found k= 5 constrained the exploration space for curiosity to be within the realm of socially appropriate behaviors. Increasing k allows for a wider variety of robot behaviors, but increases the rate of socially inappropriate behaviors.

We also tested several β values for the behavior utility function, and found 0.9 to be a good balance for executing behavior that is both socially appropriate and curious. Detailed analysis of the influence of the parameter beta is provided in the Appendix.

3.5 Evaluation of the Curiosity Learner

We conducted an evaluation of the Curiosity Learner's ability to adapt to individual customer's behaviors.

There are two learning phases for the Curiosity Learner. The first learning phase is when the Curiosity Learner (and the Appropriateness Learner) are trained on the human-human dataset (Sec. 3.4.2) – i.e. offline learning. The second phase of learning occurs in *real time*, i.e. online learning, when the proposed system interacts with a real human customer (Sec. 3.4.3). Recall that the main task of the Curiosity Learner is to predict the customer's next action $h(t+1)$ based on the current customer and shopkeeper actions $h(t), r(t)$. During the offline learning phase the Curiosity Learner learns to predict the *most frequent* customer action that followed those customer and shopkeeper actions in the training dataset. However, during the second, online, learning phase, the Curiosity Learner learns to more accurately predict the actions of the *individual* customer who is currently interacting with the robot. This is accomplished by training the Curiosity Learner on the observed individual customer's actions, such that the probability of predicting the previously observed actions increases.

The Curiosity Learner's ability to learn from the customer's individual behaviors is critical for generating more varied, interesting interactions. When the Curiosity Learner's ability to predict the customer's behavior in response to certain robot actions improves, the robot's 'curiosity' drives it to perform other actions, for which the customer's reaction is less predictable.

Thus, the evaluation described below focuses on the online learning phase. To more thoroughly evaluate the online learning phase, an additional interaction dataset was created via simulation. Subsequently, the Curiosity Learner's customer action prediction accuracy and the curiosity scores of the simulated robot actions were measured before and after online learning.

3.5.1 Simulated Interactions

We simulated interactions between a robot shopkeeper and a human customer to create a dataset for evaluation separate from the human-human dataset (Sec. 3.3), since the human-human dataset was used to train the Curiosity Learner in the first, offline, learning phase (Sec. 3.4.2). Interactions were simulated using a graphical user interface that displayed the customer and shopkeeper's locations on a map of the camera shop. A human user was able to control the customer's speech by text input and the customer's location by mouse click. The shopkeeper's actions (speech text, spatial state, and state target) were automatically selected in response to the customer's actions by the Appropriateness Learner (Sec. 3.4.3) and displayed to the user. Using the simulation tool, 15 interactions were generated by a user playing various types of customers (a professional photographer, an art student, a frequent family-vacationer, a silent browsing customer, etc.). 2 to 3 interactions were role played for each customer type. The 15 interactions contained 205 customer-robot action turns in total. The average interaction length was 13.7 turns (s.d. 2.9).

3.5.2 Simulated Evaluation Setup

The experiment consisted of comparing the Curiosity Learner's customer action predictions under two experimental conditions. Predictions were made for each of the 205 simulated customer actions, $h(t)$, given the context $h(t-1), r(t-1)$. The conditions were:

No Adaptation: The Curiosity Learner predicted the customer's actions *without online learning*.

With Adaptation: The Curiosity Learner's predictions of all customer actions $h(t)$ in an interaction i were made after sequentially online-learning from each of $h(t-1), r(t-1) \rightarrow h(t)$ in interaction i. This condition shows the Curiosity Learner's capability to learn and remember the customer's behaviors over the entire duration of the interaction.

3.5.3 Evaluation Metrics

3.5.3.1 Customer action prediction accuracy

To evaluate the effectiveness of online learning, the customer action prediction accuracy on each of the 205 customer actions was measured in both of the experimental conditions. To determine whether a predicted customer action $h'(t)$ matches the simulated ground truth customer action, the ground truth action is first matched to a customer action cluster $\hat{h}(t)$ using the nearest neighbor classifier, as described in Sec. 3.4.3, and if $h'(t) = \hat{h}(t)$, it is considered a correct prediction.

Furthermore, we looked at both whether the ground-truth-matching customer action $\hat{h}(t)$ was the Curiosity Learner's top prediction ('top-1 accuracy') and whether it was among the top 5 predictions ('top-5 accuracy').

3.5.3.2 Average curiosity scores

While the customer action prediction accuracy shows how well the Curiosity Learner learns about individual customers, the curiosity score, which is computed from the Curiosity Learner's predictions, has a more direct effect on the robot's behavior. When the curiosity score for a shopkeeper action is high, the robot is more likely to take that action. Furthermore, after observing a customer action $h(t)$ in response to a previous robot action $r(t-1)$, the curiosity score for $r(t-1)$ should decrease in the context of actions like $h(t-1)$, such that if the customer performs a similar action to $h(t-1)$ subsequently, the robot will be more likely to explore a different shopkeeper action. An example of this process is presented below in Fig. 3.11.

Average curiosity scores for both experimental conditions were computed over each of the 205 $h(t-1), r(t-1)$ action pairs in the simulated dataset. That is, each $h(t-1), r(t-1)$ was fed into the Curiosity Learner in both the *no adaptation* and *with adaptation* conditions, and curiosity scores were computed from the learners' outputs, in the form of probability distributions over the set of possible customer actions

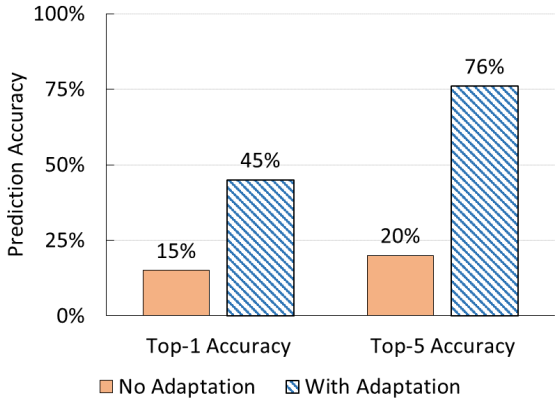


Figure 3.7: The Curiosity Learner's customer action prediction accuracy with and without adaptation to individual differences

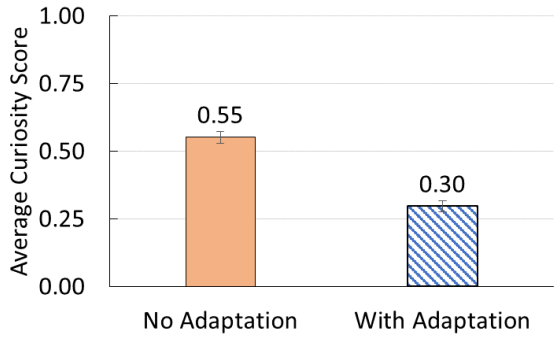


Figure 3.8: Average curiosity scores of each simulated turn with and without adaptation

$p(h(t+1)^1), \ldots, p(h(t+1)^K)$, using equation (4.5.1).

3.5.4 Evaluation Results

The evaluation's results are shown in Fig. 3.7 and 3.8.

3.5.4.1 Customer action prediction accuracy improves after adaptation

The results in Fig. 3.7 show that the Curiosity Learner with adaptation, i.e. the online learning phase – based on individual customer behavior, was better able to predict the customer's actions. The Curiosity Learner had top-1 accuracy of 45% with adaptation and 15% with no adaptation. The customer action prediction accuracy matters because when the Curiosity Learner predicts a previously observed action with high probability, this leads to a low curiosity score for robot actions that might cause the customer to respond in the same way again. This enables the robot to explore different actions, resulting in more varied, interesting interactions. For example, this is the case for robot action R3 in the example interaction shown in Fig. 3.11.

To compute the curiosity score, the top k predictions are used. In our implementation, k was set to 5. Therefore, it is also useful to evaluate how often the correct customer action is among the Curiosity Learner's top 5 predictions. The Curiosity Learner had top-5 accuracy of 76% with adaptation and 20% with no adaptation. This means that for about 76% of the instances the Curiosity Learner learned sufficiently to try performing a different, curious action if the robot were to encounter a similar situation again.

3.5.4.2 Curiosity scores decrease with adaptation

Fig. 3.8 shows how the average curiosity scores of the simulated robot actions $r(t)$ changed as a result of adaptation (with error bars showing the standard error). The average curiosity score without adaptation was 0.55 and dropped to 0.30 with adaptation. This is a consequence of learning the customer's action $h(t+1)$, as demonstrated in Sec. 3.5.4.1. When the Curiosity Learner adapts to the customer's behavior, it becomes more 'confident' in its prediction of the customer action, which is reflected in a less uniform distribution over the possible customer actions. As a result, the entropy, and thus the curiosity score, is lower. This yields a lower utility for the action $r(t)$, allowing the robot to explore other actions instead.

3.5.4.3 Runtime performance

The online learning is a very fast process. Each of the 205 simulated instances of online learning completed in an average of 55.7 ms (s.d. 33.1) when run with a GeForce GTX 980 Ti graphics card. The run-time performance during the user study, presented below in Sec. 3.6, was similar and did not delay the robot's response time.

In summary, these evaluations demonstrate that the Curiosity Learner is able to effectively learn an individual customer's behaviors through interaction. During real time interaction, the Curiosity Learner's ability to learn is critically important since curiosity scores are assigned to robot actions based on the learner's output. Thus, adaptation allows the proposed technique to vary its behavior based on what has been learned about the customer.

3.6 User Study

3.6.1 Conditions

To observe the effect of the proposed techniques during live interaction, we conducted a user study to compare the two conditions:

Curious robot: uses both the *Appropriateness Learner* and the *Curiosity Learner* for generating robot behaviors

Non-curious robot: uses only the *Appropriateness Learner*.

The experiment used a within-participants design and the order of the conditions was counterbalanced to avoid ordering effects.

3.6.2 Hypothesis and Prediction

We made the following hypotheses about the effects of our proposed techniques:

- Driven by curiosity, the curious robot will perform more actions it has not taken before, mimicking humanlike-ness. Thus, it will be perceived as being more *humanlike*.
- The curious robot will be able to adapt its behaviors to some individual customers' differences, creating opportunities for interactions to develop in diverse ways and resulting in more *interesting* interactions.
- Providing a more individualized interaction will lead to more enjoyable experiences for the customer, thus the curious robot will be perceived to have *better interactions* overall.

3.6.3 Experiment Setup

3.6.3.1 Participants

A total of 16 paid participants (10 male and 6 female, average age 34.9, s.d. 9.0) were recruited for this experiment. All of them were fluent English speakers and had no previous familiarity with the robot.

3.6.3.2 Environment

The experiment was conducted in the same camera shop setting used for the human-human data collection, with three cameras displayed in an 8m x 11m experiment space. The same sensor network was used for tracking, and the participants communicated with the robot using a handheld smartphone for speech recognition.

3.6.3.3 Robot platform

For this experiment, we used a humanoid robot with a 3-Degree-of-Freedom (DOF) head, two 4-DOF arms, and a wheeled base, capable of moving at a speed of 0.7 m/s. We implemented the proposed techniques in the robot, enabling it to autonomously generate behavior based on inputs from the sensor network and speech recognition results. The dynamic window approach (DWA) was used to avoid obstacles [30], and the speech synthesis system described in [58] was used to generate utterances.

To make the interactions more natural, idling behavior was implemented in the robot for both conditions, in which the robot makes small arm and head movements while idling, speaking, and moving [115]. Automatic head-tracking of the robot's interaction

partner was also implemented, and the robot followed the customer with its gaze during all interactions.

3.6.3.4 Procedure

Since the goal of this study was to evaluate a curiosity-driven robot, we asked the participants to role-play as a customer, and to observe and interact with the robot with the purpose of evaluating the quality of the interaction. Each participant interacted with both robots. To test the robot, we suggested the participants could ask about the same camera features twice, remain silent by playing with the camera, or acknowledge the robot with some simple utterances (e.g. "ok"). Participants freely chose when to end each interaction, by thanking the robot and leaving the shop.

As in our human-human data collection, before the start of the experiment, we explained to the participants what a typical interaction was like, asked them to become familiar with the smartphone interface, and confirmed their understanding of the instructions.

After each participant had interacted in one condition, a questionnaire was administered, and the procedure was repeated with the remaining condition (curious or non-curious). Half the participants interacted with the curious robot first then the non-curious robot, and the other half interacted in the opposite order.

3.6.4 Measurement

After each condition, participants filled out a written questionnaire, rating the following items on a 1-7 scale (1 being very negative and 7 being very positive):

1. How humanlike did the robot's behavior seem to you, considering the repetitiveness and diversity of its behaviors?
2. How interesting was your interaction with the robot?
3. Was the robot's behavior socially-appropriate for its role as a shopkeeper?
4. Overall evaluation

After the questionnaire was completed, the participants were briefly interviewed.

3.6.5 Results

3.6.5.1 Questionnaire results

To compare each rating between the curious and the non-curious robot, we conducted a paired t-test for each of the four questions, the results of which are shown in Fig. 3.9.

We verified that all of our hypotheses were supported, as the analysis found significant differences between the conditions for ratings: Humanlike ($t(15)=3.748$, $p=.002$),

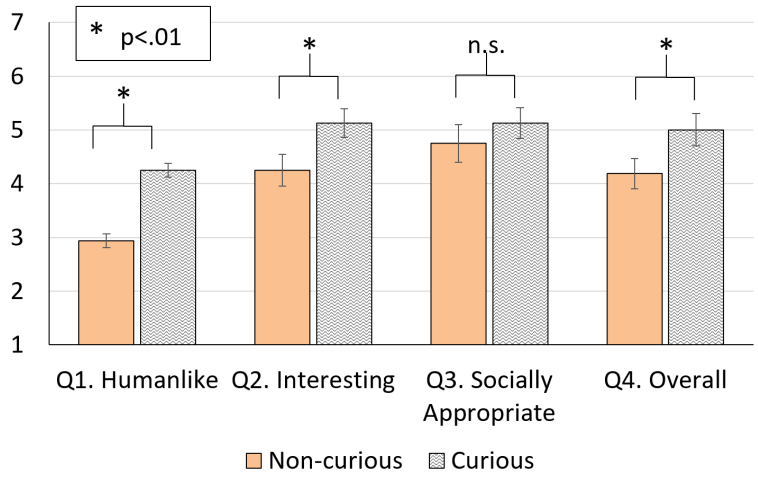


Figure 3.9: User study questionnaire responses

Interesting (t(15)=3.050, p=.008), and Overall evaluation (t(15)=3.896, p=.001). We did not find a significant difference for Socially-appropriate (t(15)=1.695, p=.111).

Thus, the results support our predictions: The curious robot was perceived to be more *humanlike* with respect to repetitiveness and diversity of behavior and more *interesting* than the non-curious robot; there was no significant difference between the perceived *social appropriateness* between two robots; and the curious robot was rated to have a *better overall interaction* than the non-curious robot.

3.6.5.2 The curious robot asks more questions

We conducted an analysis of how many questions occurred in each condition of the user study (Fig. 3.10). Each robot action in the user study logs from 10 of the experiment sessions (only 10 out of 16 participants were analyzed due to failures in recording in the other 6 trials) was labeled by an expert annotator as either *greeting*, *closing*, *inform*, *question*, *response*, or *null* (actions where the robot remained silent – e.g. waiting at the service counter or silently standing by the customer). Then, the mean number of occurrences per interaction were computed for each condition and a t-test was performed.

The only utterance type to have a statistically significant difference between the two conditions was *question*. In the *non-curious* condition, the robot asked the customer an average of 0.6 questions per interaction. In the *curious* condition, the robot asked the customer an average of 2.4 questions per interaction. Thus, the *curious* robot asked four times as many questions as the *non-curious* robot.

Some of the questions asked by the curious robot were, “What sort of pictures do you like to take?”, “What sort of camera did you have before?”, and “What kind of camera are you looking for?” This may be one reason that participants observed more humanlike diversity of behavior in the curious robot and why they found those interactions to be more interesting.

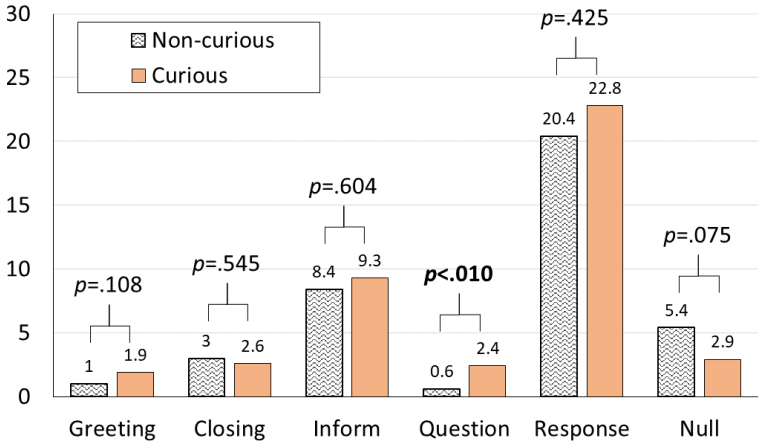


Figure 3.10: Frequency of robot utterance types. The curious robot asks significantly more questions than the non-curious robot.

3.6.5.3 Qualitative observations

During the experiment we observed that the curious robot generated behaviors with a wider variety of actions, which improved the quality of the interactions in these situations: 1) when the customer repeatedly responded to the robot with backchannels, the robot provided a variety of information rather than repeating itself, 2) when the customer was silent, the robot attempted various behaviors rather than repeating the same behavior, and 3) when the customer repeated the same question more than once, the robot was able to respond with different phrasings. Considering the third case, in a real interaction a customer may repeat himself because he did not understand what the robot said the first time. In this case, it is helpful for the robot to try rephrasing its response.

The curious robot often performed several types of behaviors: 1) rephrasing a response; 2) elaborating on basic responses by providing additional information; 3) performing proactive behaviors, such as making camera recommendations or guiding the customer to different cameras; and 4) asking the customers questions about their preferences. For these reasons, the curious robot appeared to be more engaging to the customers than the non-curious robot. Some participants stated that the curious robot had more salesman-like qualities than the non-curious robot, and many participants preferred interacting with the curious robot.

We found that the curious robot spoke more unique utterances than the non-curious robot. On average, the curious robot spoke 30.6 (s.d. 5.5) unique utterances, while the non-curious robot only spoke 21.3 (s.d. 9.0) unique utterances ($t(9)=3.274$, $p=.010$). To see if there was a difference in customer behavior, we analyzed the total number of utterances spoken by each customer. Although not significant, ($t(9)=1.336$, $p=.214$), the customers spoke a slightly higher average of 32.4 utterances (s.d. 13.3) to the curious robot, compared with 27.7 utterances (s.d. 7.3) to the non-curious robot.

3.6.6 Case Study

Using the proposed techniques, we observed that the robot's behaviors appeared to be driven by curiosity. Here we present two examples.

3.6.6.1 Example 1

Fig. 3.11 illustrates an example where the curious robot was able to continuously adapt to the customer's responses during live interaction (i.e. exploring different actions given the same customer input), rather than always using the same "default" interaction style learned from the off-line training.

At first, the customer is preoccupied with the Canon camera and yields her turn by remaining silent for a period of time. Once the robot detects a yield action (highlighted in blue), it queries the *Appropriateness Learner* to output the top 5 most appropriate robot actions, with an appropriateness value of around 0.05 for each of the five actions. For each robot action, the *Curiosity Learner* predicts a probability distribution for the top five subsequent customer actions, for which the *curiosity* value is computed. Here, Action R1 has the highest *curiosity* value of 0.567, yielding a total *utility* value of 0.205. The robot thus executes Action R1 and asks, "Can I ask what sort of pictures you take?"

The customer answers that she is looking to take pictures of family and friends, and this utterance is matched to the nearest customer action cluster. Once the robot observes this customer action, it updates the weights of the neural network in the *Curiosity Learner*. The robot then talks about the Canon camera, after which the customer ignores the robot and continues to remain silent (highlighted in brown). Given the same input, the *Appropriateness Learner* outputs the same 5 possible robot actions as before, but because the *Curiosity Learner* has been updated to reflect the customer's behavior, the customer action probabilities have changed, and the *curiosity* value for Action R1 has decreased. As a result, the robot decides to take Action R3, which now has the highest *utility* value among the possible actions, with a value of 0.301. The robot then introduces additional features of the Canon camera, "Full frame it's same size as a piece of 35 mil film that's the standard for a top end camera."

3.6.6.2 Example 2

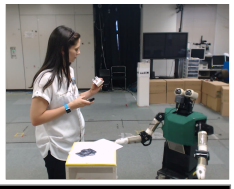
The left side of Fig. 3.12 shows the interaction history of an engaged customer, who is asking the robot many questions. In customer action C6, the engaged customer responds to the robot with a backchannel, "Okay." Based on the customer's previous actions, the Curiosity Learner has learned that he is likely to continue the conversation with any of a number of proactive actions. So, the robot responds to the backchannel phrase in an engaged way, "You might like to pick it up and try taking a few shots."

In contrast, the right side of Fig. 3.12 shows the interaction history of an unengaged customer. In this interaction the robot tries offering information or asking questions, but is answered with short, disinterested responses, e.g. "Not sure" and "I see," or is ignored by the customer. Finally, when the customer responds to the robot's offered

Interaction example

(Present Canon)

C1: *(plays with the camera and remains silent)*
R1: Can I ask what sort of pictures you take?
C2: I am looking to take pictures of family and friends.
R2: It's the top end professional camera comes in black.
C3: *(plays with the camera and remains silent)*
R3: Full frame it's same size as a piece of 35 mil film that's the standard for a top end camera.



	Possible robot actions at C1	Uti	App	Cur	Possible customer actions at $h(t+1)$: C2
R1	**(Present Canon) Can I ask what sort of pictures you take?**	**0.205**	**0.051**	**0.567**	0.507 I'm going on vacation soon just looking for a small camera 0.054 I'm just looking for a basic camera to take selfies with 0.046 oh yeah actually it was but I need the previous model 0.046 okay thank you how 0.026 I'm looking for a camera that will take automatic pictures
	(Present Canon) Full frame it's same size as a piece of 35 mil film that's the standard for a top end camera.	0.052	0.049	0.060	0.970 (Customer silent) 0.008 ok 0.002 okay alright thank you have a look around 0.002 yeah I think you're right now 0.002 how much does this cost
	(Present Canon) Some people think it's a bit heavy it weighs one kilogram but that is what you get when you buy a top end camera.	0.037	0.053	0.000	...
	(Present Canon) You Can do everything in manual mode so it will also give you software style graph that you can manually set to adjust histograms and tone within the camera manually.	0.041	0.053	0.013	...
	(Present Canon) You have full creative control it has every possible manual setting	0.034	0.048	0.001	...

	Possible robot actions at C3	Uti	App	Cur	Possible customer actions at $h(t+1)$: C4
	(Present Canon) Can I ask what sort of pictures you take?	0.006	0.054	0.001	0.959 I'm looking for a camera to take pictures of the family 0.025 I'm going on vacation soon just looking for a small camera 0.002 oh yeah actually it was but I need the previous model 0.002 I'm looking to take pictures of trains 0.001 I'm looking for a camera that will take automatic pictures
R3	(Present Canon) Full frame it's same size as a piece of 35 mil film that's the standard for a top end camera.	**0.546**	**0.051**	**0.601**	0.275 I'm looking for a camera to take pictures of the family 0.110 ok 0.080 I'm going on vacation soon just looking for a small camera 0.074 okay alright thank you have a look around 0.064 how much does this cost
	⋮				

Figure 3.11: An example interaction of the curiosity-driven robot. The columns (e.g. Uti, App, and Cur) are the respective *utility*, *appropriateness*, and *curiosity* values. The probability distribution of the next customer action (e.g. Prob C2), h(t+1) is also shown. For brevity, the predicted customer actions are only shown for the relevant robot actions and only two of the five possible robot actions are shown for C3.

Engaged Customer Interaction Example

(Waiting)	
C1:	Excuse me.
R1:	*(shopkeeper approaches customer)* Good afternoon how can I help?
C2:	How many glamor retouch effects are there on this camera?
R2:	It has 13 different artistic effect modes.
C3:	What is the price?
R3:	It's only $550.
C4	What is the sensor size?
R4	Yes it does, it has a three quarter size sensor, but the pictures of that are almost as good as the pictures from the 5d Mark 3, only a professional could tell the difference.
C5	Oh wow that's nice.
R5	there are limits on how far you can go with the shutter it will only go up to 30 seconds that's not bulb mode so depending on the effect you want.
C6:	Okay.
R6:	You might like to pick it up and try taking a few shots.

Unengaged Customer Interaction Example

(Waiting)	
C1:	*(looks at the Nikon camera)*
R1:	Good afternoon how can I help?
C2:	I'm just looking for a camera
R2:	We have a lot of different cameras for every need can I ask what soda pictures you take?
C3:	Not sure.
R3:	Let me explain the layout of the store over here we have the top end professional cameras in that camera over there sells just the body for $2000 but it does everything.
C4	I see.
R4	This one costs only $68, fits in the palm of your hand.
C5	*(ignores robot)*
R5	This lever here is the optical zoom so if you push it one way or the other just see how that goes.
C6:	Okay.
R6:	If you have any other questions please let me know.

Possible robot actions at C6	Possible *engaged* customer actions at $h(t+1)$: C7	Possible *unengaged* customer actions at $h(t+1)$: C7
(Waiting) If you have any other questions please let me know.	0.416 Uti 0.269 App 0.433 Cur 0.710 (Customer silent) 0.137 what about the sensor size 0.041 does it take a movie 0.008 okay thanks 0.007 thank you all look around	**0.154 Uti 0.269 App 0.141 Cur** 0.941 (Customer silent) 0.056 (Customer goes to Sony) 0.001 (Customer goes to Canon) 0.001 let me see 0.001 thanks goodbye
(Present Sony) You might like to pick it up and try taking a few shots.	**0.509 Uti 0.029 App 0.563 Cur** 0.518 what kind of effects does it have 0.221 what about the sensor size 0.021 excuse me 0.020 okay I see and has synthesizes and long exposures in manual settings on spreading good camera 0.015 is this the best you have in this shop	0.011 Uti 0.029 App 0.009 Cur 0.997 (Customer silent) 0.001 let me see 0.001 and how about the ISO settings 0.001 (Customer goes to Canon) 0.001 is this heavy
(Present Sony) Travel pictures family and friends to go out and take night pictures?	0.430 Uti 0.251 App 0.450 Cur ...	0.033 Uti 0.251 App 0.008 Cur ...
(Present Sony) The Sony a6000 is designed to take pictures almost as good as the top end cameras you been using, but feel how light it is it's only half a kilogram.	0.088 Uti 0.077 App 0.089 Cur ...	0.017 Uti 0.077 App 0.010 Cur ...
(Present Sony) Can I ask what kind of pictures you take?	0.406 Uti 0.034 App 0.0447 Cur ...	0.053 Uti 0.034 App 0.055 Cur ...

Figure 3.12: An example interaction of the curiosity-driven robot. Uti, App, and Cur are the respective *utility*, *appropriateness*, and *curiosity* values. The probability distribution of the next customer action (e.g. Prob C2), $h(t+1)$ is also shown. For brevity, the predicted customer actions are only shown for the relevant robot actions.

information with "Okay," the Curiosity Learner, having learned that the customer is likely to give some short response or to remain silent, assigns the highest curiosity score, 0.141, to the action "If you have any other question, please let me know" with the robot returning to the service counter. This action is predicted to possibly result in the customer going to explore the other cameras in the shop, so in that way it is the most curious action. Conversely, had the robot selected the same action as for the engaged customer, it would almost certainly have been met by silence from the customer.

3.7 Discussion

3.7.1 Curiosity and Individual Differences

We found that the 'curious' robot was able to adapt its behaviors to some individual customer differences (e.g. interested versus uninterested customers), rather than always using the same default behaviors it learned from the off-line training. For example, when an uninterested customer continued to ignore the curious robot for some time, the robot would often go back to the service counter, saying "I will be at the service counter if you need any more help." While this was not the most proactive, salesman-like behavior, it had the highest curiosity value for that particular situation, due to the fact that the robot had 'lost curiosity' (i.e. the entropy-based curiosity score decreased) about previous actions (e.g. presenting features) since those actions did not elicit any unanticipated customer responses. In contrast, when the curious robot was interacting with an interested customer who had many questions, the robot would usually continue answering the customer's questions and would not leave the customer alone. Given the same situation, the non-curious robot would simply continue to respond with the same "default" behavior regardless of whether the customer was interested or uninterested, potentially resulting in a less ideal interaction than if it had adapted to the individual's needs.

3.7.2 Perception of Curiosity

The 'curious' robot can only exhibit behaviors that are perceived as curious if such behaviors occurred in the human-human dataset, from which the robot learns. The camera shop interaction scenario, on which we demonstrated our proposed system, contains many curious behaviors since the shopkeeper tried to learn about the customers. For example, *questions* are one type of behavior performed by the human shopkeeper that the robot learned to imitate. When the robot asks the customer a question, it is naturally perceived as being curious. If, however, the proposed system were to be applied to a scenario in which the target human is not curious, it is unlikely that the robot would perform actions that exhibit curiosity.

However, it is possible that a robot trained on a dataset without curious behaviors can still learn about the humans it interacts with. This is because, at a fundamental level, the mechanism that drives the robot's behaviors will always result in robot actions that

lead to uncertain human responses, such that the robot can learn more about the human. For example, sometimes the shopkeeper remaining silent and letting the customer take initiative creates a greater opportunity for the shopkeeper to learn about the customer, even though this might seem like a passive, uninterested behavior. Thus, there may be some benefit to applying the curious system to training datasets regardless of the quantity of curious behaviors they contain.

In our user study, we naturally wondered whether the attribute of 'curiosity' of our robot was directly perceivable by the participants. Our interviews with the participants showed that some did perceive 'curiosity' in the robot, while other participants could not tell the difference. Indeed, one common reason for the perception of 'curiosity' in the robot was that the robot asked questions (e.g. "what sort of pictures do you like to take?"), which is one quantitative aspect of curiosity [69, 117]. Our *Curiosity Learner* occasionally selects a robot utterance that is a question, and questions often lead to more variety in customer behavior. Note that our system does not have semantic understanding of which utterances constitute questions. For future work, it would be interesting to incorporate semantic understanding to better model curiosity for a conversational robot.

3.7.3 Generalizability

In this study, participants were instructed to focus only on camera-related conversation, which is not fully natural. It is uncertain how the proposed techniques could generalize to more natural conversation with a wider variety of topics. Currently, one limitation of our approach is that the robot cannot explicitly act on learned information, e.g. for the purpose of goal-directed behavior. However, we anticipate that additional mechanisms, such as incorporation of time series data, would enable a curious robot to exploit information it had learned about the customer earlier in the interaction (e.g. "Are you planning to travel soon?"), and pursue specific goals (e.g. "This camera is great for travel").

In principle, we expect that the proposed approach can be applied to any domain characterized by repetitive, formulaic actions but also containing opportunities for individualized interaction (e.g. a waiter, receptionist, or travel agent). While the simple *Appropriateness Learner* can learn the repetitive behaviors, the *Curiosity Learner* can discover which behaviors are likely to lead to individual variation in customer responses. By guiding the interaction towards these behaviors, the curious robot creates opportunities for interactions to develop in diverse ways, opening up paths in the dialog that have the potential to branch out according to an individual's interests or needs. For example, a conversation with a curious robot that generates the curious action, "What kind of pictures do you like to take?" may lead to topics relating to the customer's hobbies, and end with the shopkeeper recommending a camera. In contrast, the non-curious robot would never have even asked the question in the first place.

3.8 Conclusion

In this work we have presented a curiosity-based system for generating interactive behavior for a social robot. To the best of our knowledge, this study is the first to apply curiosity-based learning to this domain, to drive adaptation and individualization of dialogue. Our curious robot initially learns socially-appropriate behavior by imitation from offline data, then continues to learn online, during live interaction with humans. The system adapts to customers' reactions in real time by choosing its own actions in order to explore and satisfy its curiosity about the customers' individual differences. In a user study we found that participants rated the curious robot to be significantly more humanlike with respect to repetitiveness and diversity of behavior, interesting, and better overall in comparison to a non-curious robot. The proposed techniques contribute a step towards advancement in the area of intrinsic motivation and curiosity-based learning for social robots. In future, it would be interesting to explore the application of the proposed technique on larger datasets and to other scenarios.

3.9 Appendix

3.9.1 Analysis of the Curiosity Parameter

The curiosity parameter β is intended to control how curious the robot is when choosing an action. Specifically, it determines how much weight the robot places on the curiosity scores and social appropriateness scores of possible robot actions. We analyzed the effect of the curiosity parameter's value on the robot's behavior by comparing the robot's output action before and after adaptation at various curiosity parameter values.

3.9.1.1 Exploration rate metric

To measure the effect of the curiosity parameter value on the robot's behavior, we define *exploration rate* as the percent of instances where the robot's action in response to $h(t)$ without adaptation, $r_{no\ adaptation}(t)$, is different from its response to $h(t)$ after adaptation, $r_{no\ adaptation}(t)$. Formally,

$$exploration\ rate = \frac{\sum_{n=1}^{N}\sum_{t=1}^{numturns_n}\left(r_{no\ adaptation}(n,t) \neq r_{with\ adaptation}(n,t)\right)}{\sum_{n=1}^{N}\sum_{t=1}^{numturns_n}1} \quad (3.9.1)$$

where N is the number of simulated interactions ($N = 15$), and $r_{no\ adaptation}(n,t)$ and $r_{with\ adaptation}(n,t)$ are the robot responses to customer action $h(t)$ at timestep t in interaction n. The *no adaptation* system used the Curiosity Learner before learning individual customer behaviors, and the *with adaptation* system used the Curiosity System that had learned about individual customer behaviors (as described in Sec. 3.5.2).

Thus, a high exploration rate means the robot changes its behavior more in response to learning about individual customer behaviors.

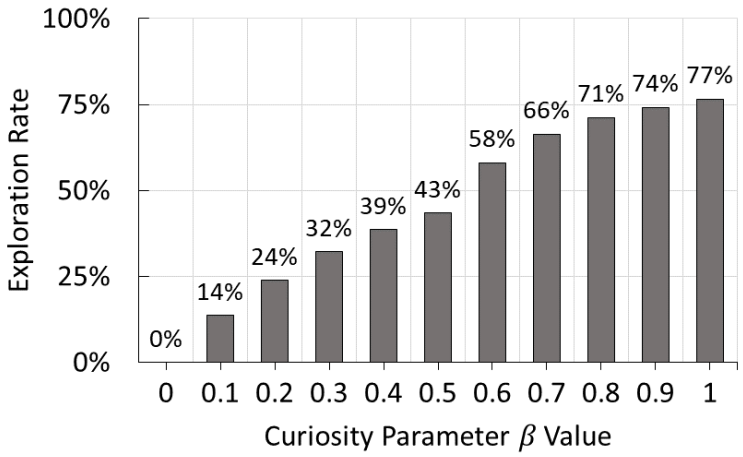


Figure 3.13: The effect of the curiosity parameter on how frequently the robot changes is response to customer actions after adaptation

3.9.1.2 Effect of curiosity parameter on exploration rate

The exploration rate is the most direct method of measuring the effect of learning individual customer behaviors on the robot's behavior. The results in Fig. 3.13 show that the effect of adaptation on the robot's actions increases as the curiosity parameter β increases. When $\beta = 0$ adaptation has no effect on the robot's behavior because in this case the system does not use the output of the Curiosity Learner. But, when $\beta = 1.0$ the robot explores a different action in 77% of the instances after adaptation than it would have performed before adaptation. This demonstrates the effectiveness of the Curiosity Learner in adapting the robot's behaviors to individual customer behaviors.

Chapter 4

Modeling Time-Persistent Interaction Phases for Ambiguity Resolution

Abstract –This work presents a learning-by-imitation technique that learns social robot interaction behaviors from natural human-human interaction data and requires minimum input from a designer. To solve the problem of responding to ambiguous human actions, a novel topic clustering algorithm based on action co-occurrence frequencies is introduced. The system learns human-readable rules that dictate which action the robot should take based on the most recent human action and the current estimated topic of conversation. The technique is demonstrated in a scenario where the robot learns to play the role of a travel agent. The proposed technique outperformed several baseline techniques in qualitative and quantitative evaluations. It responded more accurately to ambiguous questions and participants found it was easier to understand, provided more information, and required less effort to interact with.
Index Terms: Human-robot interaction, imitation learning, interaction structure, spoken dialog system, unsupervised learning

4.1 Introduction

Presently, social robots show potential in the roles of elder care, personal companions, hotel concierges, and in day-to-day interaction [65, 102, 107], and with the cost-effectiveness of automation, this trend is likely to continue. However, one of the difficulties of introducing robots to new domains is the creation of social interaction logic, which dictates how the robot behaves and interacts with people. It is tedious for an interaction designer to create all the behaviors for a robot by hand, and it is an incredibly challenging task to anticipate all the varieties of ways that humans may behave in a social interaction.

All these problems can be addressed through an imitation learning based approach for the development of robot interaction logic, as demonstrated by Liu et al. [78, 79]. Natural human-human interaction data can be collected with sensor networks in target environments, like stores and offices. Machine learning algorithms can then be applied

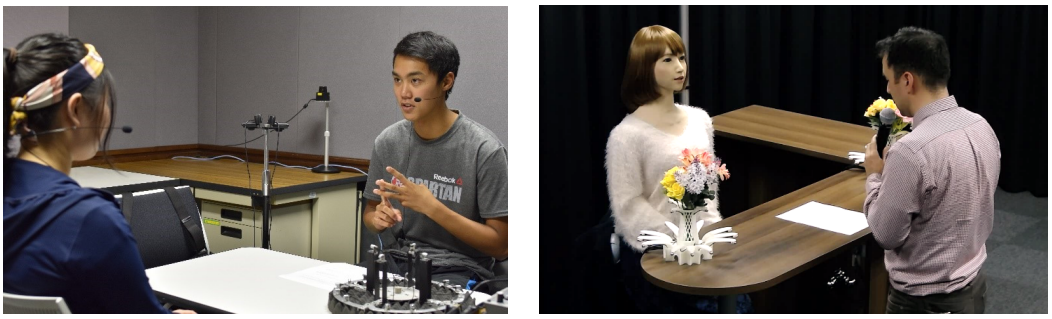


Figure 4.1: Left) Human-human data collection in travel agent domain, Right) After training the system the robot replaces the human travel agent.

to this data to train a robot to take the place of one of the humans by imitating their behavior. For example, by passively collecting data from natural interactions in a travel agency, a robot could learn how to become a travel agent (Fig. 4.1).

With an imitation learning approach it is possible for a robot to learn the correct responses to *unambiguous* human actions. However, it is much more difficult to learn how to respond correctly to context-dependent, *ambiguous* actions. For example, consider utterances like "How much is it?", which do not explicitly state the *topic of conversation*. In some special cases, it is possible to infer the relevant context from features directly observable through the robot's sensors, such as in a camera shop scenario [78], where the location of the robot and customer provided information about which camera was being discussed. More generally though, the relevant context is *hidden*, i.e., not directly observable.

To explore the problem of hidden context, in this work a travel agency scenario is presented, where the topics of conversation (travel packages) are abstract entities, which are not directly observable from the sensor data. Since the topic of conversation is hidden, some additional information about the state of the interaction must be modeled in order for the robot to form correct responses to ambiguous customer questions.

A second important problem of learning social robot interaction logic is interpretability [40]. Previous approaches to data-driven human-robot interaction, such as [78], use models that are not understandable by a human designer. However, interpretability is a desirable characteristic for several reasons. For example, in the case that the machine learning algorithm makes a mistake, it is advantageous for a human behavior designer to fix any errors in the learned interaction logic by hand. Furthermore, in many fields, such as health, finance, and defense, where the decisions of a robot may have significant consequences, it is imperative to understand the reasoning behind those decisions [86].

In this work, the problems of hidden context and interpretability are solved by modeling the *structure* of the interaction itself and learning human-readable robot interaction rules. With the proposed technique, the topics of conversation are automatically discovered by clustering speech actions observed in human-human training examples, based on action co-occurrence frequency. Then, a topic state estimator is trained to

infer the topic of conversation at each turn in a real-time interaction. Human-readable interaction rules are learned based on speech action transition probabilities and the inferred topic states in the training data. These rules dictate how the robot should respond in real time based on the customer's previous action and the current topic state.

4.2 Related Work

4.2.1 Data-Driven Robot Interaction Logic

In data-driven approaches to creating robot interaction logic the robot imitates the outward behaviors observed in human-human, or teleoperated, example interactions, without requiring a model of the world or input from a human designer.

One popular data-driven approach is to crowd-source large amounts of human-human interaction data in virtual worlds, from which to train a virtual agent or robot [9, 92, 93]. These works differ from ours since their data was collected through a virtual world, which was not susceptible to the sensor noise present in real-world interactions.

A second approach is to collect human-human interaction data and manually annotate it for supervised learning, such as [1] for a robot tutor. In comparison, our proposed approach does not rely on expensive manual annotation of the data.

A third approach is to learn from unlabeled human-human interaction data via unsupervised learning. Ref. [78] used human action cluster IDs as labels in a supervised learning framework to train a robot shopkeeper. Their system could only use information *directly observable* from the sensors, so it cannot deal with otherwise unresolvable human actions, e.g. ambiguous questions that depend on topic of conversation.

4.2.2 Topic in Dialog

A main focus of this work is to model the *topic of conversation* as it changes over the course of an interaction for the purpose of generating appropriate robot actions.

Latent Dirichlet Allocation (LDA) is a statistical approach to discovering topic based on word frequency and co-occurrence [6]. Methods based on LDA are popular unsupervised techniques for discovering the topics in sets of large documents [50, 130]. However, in contrast to the *topic of large documents*, the *topic of conversation* changes dynamically over time, so these approaches are not directly applicable to our problem.

Linguistic studies of discourse introduced the concepts of *focus of attention* and *attentional state*, which are more closely related to the *topic of conversation* [38, 39]. However, these theories rely on syntactic parsing and identification of individual words, so they are not directly applicable to situations with many speech recognition errors.

Probabilistic graphical models such as Hidden Markov Models (HMM) have been common methods for tracking states that evolve over time [90]. For example, [98] uses a dynamic Bayesian model to discover the underlying states of multiparty conversations, and [8] uses an HMM to discover interaction structure of tutoring sessions. These

works mainly focus on descriptive analysis of data, and do not use their discoveries for generating robot or agent behaviors.

Ref. [84] presented a topic tracking system for human-robot dialog which discovers the topics of conversation automatically from interaction data based on word co-occurrences however, they did not formally evaluate it to demonstrate that it works.

Ref. [71] presented a system for automatically estimating joint attention state from visual/auditory input data, comparable to our topic state estimation approach (Sec. 4.5.3).

4.2.3 Dialog Acts

Dialog acts are abstract representations of actions in dialog indicating *intent*, such as *greeting*, *request*, and *yes-no question* [120]. Unsupervised learning has been applied to automatically discover dialog acts from lexical and contextual features of utterances [27, 47]. Ref. [27] introduced a Markov random field based clustering method to discover dialog acts from transcripts of tutoring sessions. The goal of *Speech Clustering* (Sec. 4.3.4) is similar to unsupervised dialog act learning, but finds actions at a lower level of abstraction than is typical of dialog acts.

Adjacency pairs are pairs of dialog acts from alternating speakers that frequently co-occur, such as *question→statement* [7, 88, 105]. The *Robot Action Predictor* (Sec. 4.4.2) works on the basis of adjacency pairs of speech clusters that are extracted from human-human interaction examples.

4.2.4 Coreference Resolution

Modeling hidden context so the robot can answer ambiguous questions is related to *coreference resolution* (CR) – the process of determining whether two referring expressions refer to the same entity – and *anaphora resolution* – the process of determining how a previously mentioned term affects the meaning of a related term mentioned later [54].

The state-of-the-art CR system is an artificial neural network trained on manually annotated text and manually transcribed speech [20]. Ref. [20] presents an approach adapted for online chat dialog. However, CR systems have not been designed to be reliable against speech recognition errors, typical in human-robot interaction. In contrast, our proposed approach to answering ambiguous questions does not rely on manually annotated data and is robust to some speech recognition errors.

4.2.5 Dialog Systems

Some spoken dialog systems use data-driven approaches to learn dialog policies from unlabeled datasets, such as Twitter [100, 128]. However, these approaches do not model the state, i.e. topic of conversation.

Frame-based dialog systems keep track of the dialog state by tracking a set of slots and values, representing the user's goals [134]. Many approaches to state tracking, including rule-based systems, graphical models, and artificial neural networks, have been evaluated on manually labeled benchmark datasets [43]. The state of the art uses a recurrent neural network [44]. The most successful methods rely on hand-annotated corpora, so they are not suitable for rapid deployment into new domains.

In non-frame-based approaches, tree-structured ontologies [53] and Wikipedia article categories [10] have been used to represent topics of conversation. However, these approaches depend on humans to specify domain-specific knowledge or to annotate training data. Thus they are also troublesome to apply to novel domains. Our objective is to discover the topic of conversation automatically, without human input.

4.3 Data Collection

The proposed method uses human-human interaction data to train a robot to perform socially appropriate behaviors in face-to-face interactions with a human.

4.3.1 The Travel Agent Scenario

The proposed system is applicable to any domain in which the participants' actions are highly repeatable, but for demonstration a travel agent interaction scenario was chosen to train and test the system. In the scenario, a customer enters a room, approaches a table with the travel agent, and they proceed to converse about the available travel packages until the customer decides whether or not to purchase one.

Three travel packages were created for the travel agent and customer to talk about: a trip to the Sahara Desert, a trip to London, and a boat cruise along the coast of Antarctica. Additionally, each travel package had six attributes: destination, duration, price, what is included in the package, what there is to do on the trip, and who else will be going on the trip (other tourists, a desert survival guide, etc.).

4.3.2 Data Collection

In the data collection procedure, 6 fluent English speaking participants (4 male, 2 female, mean age 24.3, s.d. 3.1) took turns playing the roles of a travel agent and customer. The person in the travel agent role was provided with a listing of travel packages and their attributes. In order to collect a variety of utterances, the customer was instructed to take turns acting like three different customer types: a poor student looking for a budget vacation, an adventurous person looking for a crazy vacation, and an undecided person who simply wants to collect information. The travel agent was instructed to greet the customer, provide information about the travel packages, and refer them to checkout if a decision was made. An example data collection interaction is shown in the supplementary video.

4.3.3 Data Preprocessing

Inter-channel noise suppression and speech segmentation were first applied to identify the segments of speech in the audio data. Next, the speech segments were passed through a voice-to-text module using the Google Speech API. Lastly, a simple turn-taking model was applied, in which consecutive utterances by the same speaker with no intervening utterances by the other speaker were concatenated together. These steps result in a series of alternating customer text / travel agent text utterances.

Audio was recorded from 192 interactions with 2481 customer / travel agent utterance pairs in total, which is a comparable quantity to other datasets for training robots [9, 75, 82]. The average number of turns in each dialog was 12.9 (s.d. 3.5). The average number of words in each utterance was 9.3 (s.d. 5.8) for customers and 17.3 (s.d. 12.7) for travel agents. The automatic speech recognition (ASR) word error rate on a representative subset of 500 utterances was 23%. The average length of the interactions was about three minutes.

4.3.4 Speech Actions

In human-robot interaction there are two challenges that make textual-analysis-based dialog approaches impractical. First, the input is noisy: One HRI study reports that a speech recognition system that performed with 92.5% accuracy in 75dBA noise achieved only 21.3% accuracy in a real-world environment [116]. Second, participants in our studies often speak casually, with sentence fragments and bad grammar, hindering extraction of features based on grammatical structure.

Clustering utterances into *speech clusters* based on simple lexical features (n-grams) aids in mitigating these challenges. By grouping noisy utterances together with other similar utterances the intended meaning can be recovered by looking at the other members of the speech cluster.

The speech clusters represent the common *speech actions* in the human-human interactions.

4.3.5 Speech Vectorization

Before clustering the speech, each customer and travel agent utterance was vectorized. The utterance vectors consist of n-grams (uni-, bi-, and tri-grams) of word stems and keywords. Word stemming was done via Python NLTK's WordNet based lemmatizer[1] and keywords were extracted using AlchemyAPI[2]. The customer utterance vectorization had 2577 dimensions and the travel agent utterance vectorization had 5457 dimensions.

[1] http://www.nltk.org/

[2] http://www.alchemyapi.com/

Speech Cluster ID	Typical Utterance (Medoid)	Speech Cluster Size
2001	Hello.	61
3050	Hi there, how can I help you?	30
3004	Will ring you up at the next counter.	55
2006	Can you tell me more about the Sahara Desert?	40
2015	Okay, how much is that one?	40
3008	That is $3000.	49
2048	Okay, I guess I'll take the trip to London.	25

Table 4.1: Examples from the space of learned actions

4.3.6 Speech Clustering

The Dynamic Tree Cut hierarchical clustering algorithm was used on the utterance vectorizations to find speech clusters [70]. Customer and travel agent utterances were clustered separately since their actions did not frequently overlap. A second pass of automatic processing was performed to remove noisy clusters, which had very high intra-cluster distances, and reassigned the utterances they contained to the remaining clusters using a nearest neighbor based approach. If an utterance was farther than an empirically set distance from the nearest centroid, it was not reassigned to a cluster.

In the end, the clustering procedure yielded 113 customer clusters and 58 travel agent clusters, containing 82% of the utterances in the training data. Due to the high word error rate and the presence of non-repeatable utterances (e.g. off-topic utterances) 12% of the utterances were not clustered. Regardless, the speech clustering procedure successfully revealed the highly repeatable actions which the proposed technique aims learn. Some examples from the space of learned actions are shown in Table 4.1. Furthermore, Table 4.2 shows an example of one customer speech cluster, illustrating that the members of speech clusters include similar utterances, but with some natural variation and speech recognition errors. These speech clusters defined the 'actions' of the system.

Note, this clustering procedure splits some clusters that should ideally be merged, namely clusters of functionally-equivalent but lexically-dissimilar utterances (e.g. "How much does it cost?" and "What is the price?"). The utterance features and clustering distance metric are arbitrary, thus they could be replaced with improved techniques in the future.

Since the robot needs to be able to speak an utterance corresponding to each travel agent speech cluster, a *typical utterance* (bottom of Table 4.2) was selected for each by finding the medoid utterances, i.e., the most similar utterance to all other utterances in the speech cluster, using the Levenshtein distance metric normalized for utterance length. Since complete utterances with few ASR errors tended to share the most similarities with other utterances in the same cluster, *typical utterances* tended to be well formed and easy to understand.

Travel Agent Speech Cluster 3047	
Contents (ASR results)	The camels that you would be traveling up in your camel caravan.
	Look up the camels.
	Camel selfies.
	I'll be crossing by camel caravan.
	It will last for 14 nights and you'll be traveling by camel caravan.
	Well you could you'll be traveling by camel caravan I should say.
	Well you'll be with a lot of camels and five four other two arrests.
	⋮
Typical Utterance: "You be traveling by camel Caravan."	

Table 4.2: Example of a speech cluster

4.4 Action Prediction

Two of the three main components of the proposed system are the *Utterance to Speech Action Matcher* and the *Robot Action Predictor* (Fig. 4.2). Preliminary action prediction results using only these components demonstrate the necessity of incorporating topic of conversation to deal with ambiguity.

4.4.1 Utterance to Speech Action Matching

In the proposed system, a customer's action is recognized by vectorizing their utterance text and matching the vectorization to one of the known customer speech actions. Matching using a nearest centroid classifier with the cosine distance metric performed well empirically, achieving 91% utterance-to-speech-action matching accuracy in a 10-fold cross validation.

4.4.2 Robot Action Prediction

A robot system must decide which action to take next. In the proposed system, this *robot action prediction* is accomplished by learning a set of *human-readable* interaction rules from the sequences of speech action IDs in the human-human interaction training data. Action (speech cluster) IDs (Table 4.1) are discrete, symbolic representations of abstract actions.

More specifically, the set of interaction rules consists of *IF customer action, THEN robot action* style rules based on customer action to travel agent action transition probabilities computed from the human-human data. A rule is created for each customer action by finding the most likely travel agent action to follow using (4.4.1), where a_C is the preceding customer action, A_{TA} is the set of all possible travel agent actions, a_{TA} is the robot's action (a travel agent action), and $P(a|a_C)$ is the transition probability from

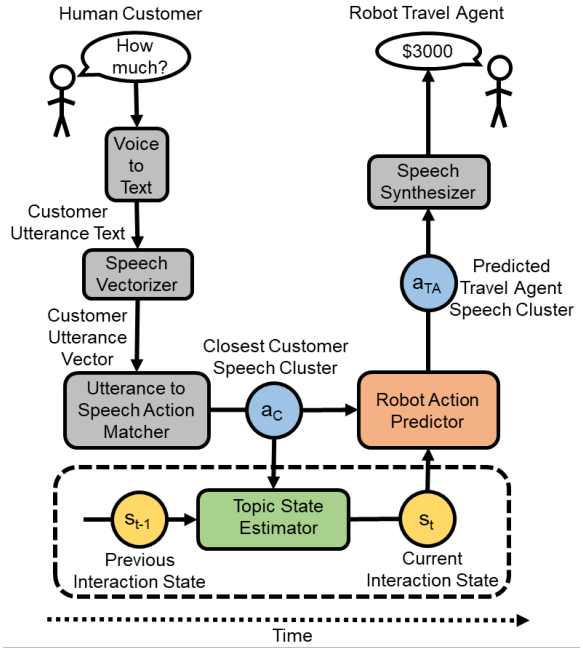


Figure 4.2: System runtime diagram. The topic module is within dashed lines.

a_C to a.

$$a_{TA} = \underset{a \in A_{TA}}{\operatorname{argmax}} P(a|a_C) \tag{4.4.1}$$

Since the goal of the system is to learn repeatable behaviors, customer action to travel agent action pairs that occurred only once in the training data were pruned.

During operation of the system, after predicting the action a_{TA}, the robot speaks the *typical utterance* (bottom of Table 4.2) of the corresponding speech cluster.

4.4.3 Preliminary Evaluation and Discussion

The system was evaluated on the 192 human-human trained interactions using hold-one-out cross-validation and a human evaluator categorized the system's responses to each customer utterance as either *correct* or *incorrect*. Most importantly, the system response correctness to *ambiguous* and *unambiguous* customer actions was examined separately.

The system learned many correct interaction rules, but some incorrect rules (Table 4.3). In particular, rules with ambiguous customer speech actions were mostly incorrect. In Table 4.3, The rule for the ambiguous customer action 2017 with *typical utterance* "Okay, and how long is that trip?", only contains a single predicted travel agent action, "That is 10 days". This is the correct response for the Antarctica travel package, but it would be incorrect for either of the other two travel packages.

Many customer utterances are ambiguous with respect to which travel package is under discussion, so predicting the robot's action based on the customer's action alone

Input Customer Action	Customer Action Typical Utterance	Predicted Travel Agent Action
2001	Hello.	Hi there, how can I help you today?
2017	Okay, and how long is that trip?	That is 10 days. (Correct only for Antarctica)
2022	Okay, I think I'm going to go with the London trip.	Okay, that's a great choice.

Table 4.3: Examples of rules learned by the system

is insufficient. In fact, only 39% of the system's responses to ambiguous customer actions were correct, compared to 66% percent of the responses to unambiguous customer actions.

4.5 Topic Estimation

To address the problem of ambiguous utterances, a topic state estimator was incorporated to model topic of conversation (Fig. 4.2). Observations of the interaction data motivated the design of a novel topic clustering algorithm. The topic clusters are used to estimate the topic of conversation in real time so the robot can respond to ambiguous customer actions.

4.5.1 Interaction Structure

The core of each training interaction consisted of the customer asking questions about the travel packages. Customers tended to ask several questions about one specific travel package at a time, then ask about another travel package. They typically asked about at least two of the packages per interaction. Interactions usually opened with an exchange of greetings and introduction of the travel packages, and closed with the travel agent thanking the customer.

These observations indicate that there are topics which constitute phases of a conversation, e.g. *London*, *Desert*, and *Antarctica*. In fact, the proposed algorithm (Sec. 4.5.2) discovered two other conversational phases, corresponding to *Opening* and *Closing*, although this was not initially anticipated. Herein the term *topic* encompasses not only the target topics but also these additional two conversation phases.

Visualizing the interaction sequences helped us to understand their structure. The background of Fig. 4.3 shows the interaction structure of several interaction sequences, as described above. Each box represents a speech action, and those speech actions which are uniquely associated with a single topic are color-coded. Speech actions that are not associated with only a single topic (i.e. ambiguous) are white. Actions associated to the same topic typically occur together in 'runs' where the topic of conversation

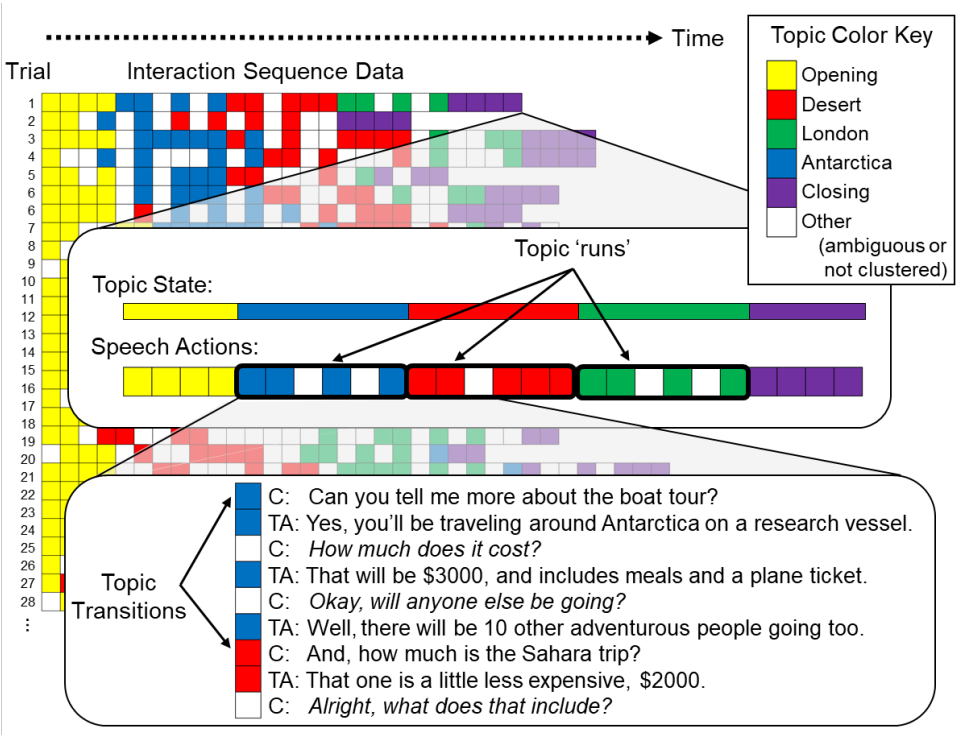


Figure 4.3: Background) The training data, with boxes representing speech actions annotated with topic; Middle) A single interaction sequence, showing topic and topic state; Foreground) A topic 'run' containing ambiguous customer utterances (ambiguous utterances are italicized

remains the same. These topic runs are sometimes interspersed by ambiguous actions (foreground of Fig. 4.3, with the ambiguous utterances italicized).

4.5.2 Topic Clustering Algorithm

Motivated by the observation that speech actions uniquely associated with a single topic often occur together, a clustering algorithm was designed based on action co-occurrence to automatically discover the structural patterns, including topics of conversation, in the training data. The end goal of clustering actions into topic clusters is an action-to-topic mapping that can be used to disambiguate customer speech.

The objective is to find sets of actions (topic clusters), whose members frequently co-occur with each other but not with members of any other set. A topic cluster should contain actions associated with that topic, such as unambiguous actions (e.g. "Tell me about the London trip?", "It includes five nights in a hotel"), and exclude actions associated with other topics, such as ambiguous questions (e.g. "How long is that one?", "What's included in the price?") and backchannels (e.g. "Okay").

To quantify the level of co-occurrence between each pair of speech actions, subsequences of length 2 to 5 actions were generated from the original interaction se-

Supp	Support matrix
T	Set of topic clusters τ ϵ T
τ	Set of actions a ϵ τ
Input: Supp	
1.	Supp' ←filter(Supp, θ_1)
2.	T←initialize_topic_clusters(Supp', θ_2)
	While stop_condition_not_met():
3.	topicFits←compute_action_to_topic_fits(Supp', T)
4.	**Add** action α to topic τ that meets Fit criteria using θ_3 and θ_4.
5.	topicFits←compute_action_to_topic_fits(Supp', T)
6.	**Remove** all actions from topic clusters that do not meet Fit criteria using θ_3 and θ_4.
	Return T

Table 4.4: The topic clustering algorithm

quences and the *support* [3] between pairs of actions was calculated using (4.5.1), where (a_i, a_j) is the action pair, H is the set of all interaction subsequences, and h is a generic interaction subsequence. Thus, the support of action pair (a_i, a_j) is equal to the proportion of interaction subsequences that contain that pair.

$$\text{Supp}(a_i, a_j) = \frac{|\{h \in \text{H};\ (a_i, a_j) \subseteq \text{h}\}|}{|H|} \tag{4.5.1}$$

With (4.5.1) a support matrix, $Supp$, was generated representing the support of all possible pairs of actions, $Supp \in \mathbb{R}_{\geq 0}^{|A_C \cup A_{TA}| \times |A_C \cup A_{TA}|}$, where A_C is the set of all customer actions and A_{TA} is the set of all travel agent (robot) actions.

The proposed topic clustering algorithm (Table 4.4) takes the support matrix $Supp$ as input and returns a set of topic clusters T. A topic cluster τ, $\tau \in$ T is defined as a subset of speech actions a, $a \in A_C \cup A_{TA}$. The topic clusters τ are mutually exclusive (an action can only be in one topic cluster).

The first step of the algorithm (Step 1 in Table 4.4) is to filter out action pairs with support below a threshold, θ_1, since actions that co-occur infrequently tend to only co-occur at all due to random noise or speech clustering errors.

Next, the algorithm initializes the topic clusters by creating seed clusters from the action pairs with very high support, using a higher threshold, θ_2 (Step 2). Action pairs with high support frequently co-occur, so they probably belong to the same topic. This step is accomplished by selecting pairs of actions with support above θ_2 and then performing connected components analysis to find groups of connected actions.

Subsequently, the topic clustering procedure, which was inspired by iterative and density based methods for graph clustering [109], consists of two subroutines that iterate until the algorithm converges (i.e. a cycle of repeating states is detected) or a maximum number of iterations is reached (since a cycle could be arbitrarily long). One

subroutine adds actions to topic clusters and the other removes actions from topic clusters.

Before each of the subroutines is run, the *Fit*, defined in (4.5.2), of each action to each topic cluster is computed (Steps 3 and 5), which indicates how well an action a belongs in a topic cluster τ. This metric works by looking at the proportion of support of the action a with each of the actions in the topic cluster τ to support of that action with all actions in all topic clusters T. If an action has a high degree of co-occurrence with most of the actions in a certain topic cluster but not with actions in any other topic clusters, then it probably uniquely belongs to that topic.

$$\text{Fit(a}, \tau) = \frac{\sum_{a_\tau \in \tau} \text{Supp}(a, a_\tau)}{\sum_{\tau \in T} \sum_{a_\tau \in \tau} \text{Supp}(a, a_\tau)} \tag{4.5.2}$$

During the *Add* subroutine (Step 4), all actions a that meet two criteria are added to the topic cluster they best fit, τ_{best}. First, in order to assign actions to their best-fit clusters the action a should match to τ_{best} above a *Fit* threshold, θ_3. Second, to exclude ambiguous actions, the action a should fit τ_{best} at least θ_4 times better than any other cluster. The second subroutine, *Remove* (Step 6), removes all actions from topic clusters that do not meet the above criteria.

The values for θ_1 and θ_2 were chosen by looking at the support for each action pair, which has a long tail distribution. θ_1 was set to 6 in order to filter actions at the end of the long tail and θ_2 was set to 90, a value well above the elbow. θ_3 and θ_4 were chosen arbitrarily. Values of 0.4 and 3 were found to work well, respectively.

When run on the human-human interaction training data, the algorithm converged after 154 iterations and discovered 5 topic clusters. In an experiment with 108 runs with various parameter values, all of them converged. The minimum, maximum, and average number of iterations before the algorithm converged were 93, 251, and 153 (s.d. 28). An average run took 10.0 seconds (s.d 1.0). The maximum cycle length was 12 iterations.

Of the 171 total actions, the topic clustering algorithm clustered 53% (90) of them and left out the remaining 81. Based on manual topic annotations, 75% (129) of the 171 actions were clustered into the correct topic cluster, or correctly left out (actions labeled *Other*). Many actions that were incorrectly clustered into the wrong topic or excluded were not well-represented in the training data or contained speech clustering errors. However, among the 90 actions that fell into topic clusters, 90% (81) of them were in the correct topic cluster. Thus, the clusters successfully identified the relevant topics.

4.5.3 Topic State Estimation

In each interaction there is an underlying *topic state* (middle of Fig. 4.3), which is persistent over time and represents which of the topics discovered by the topic clustering algorithm the conversation is about at each speaker's turn. This topic state is not directly observable from the actions alone, since there are many ambiguous actions and

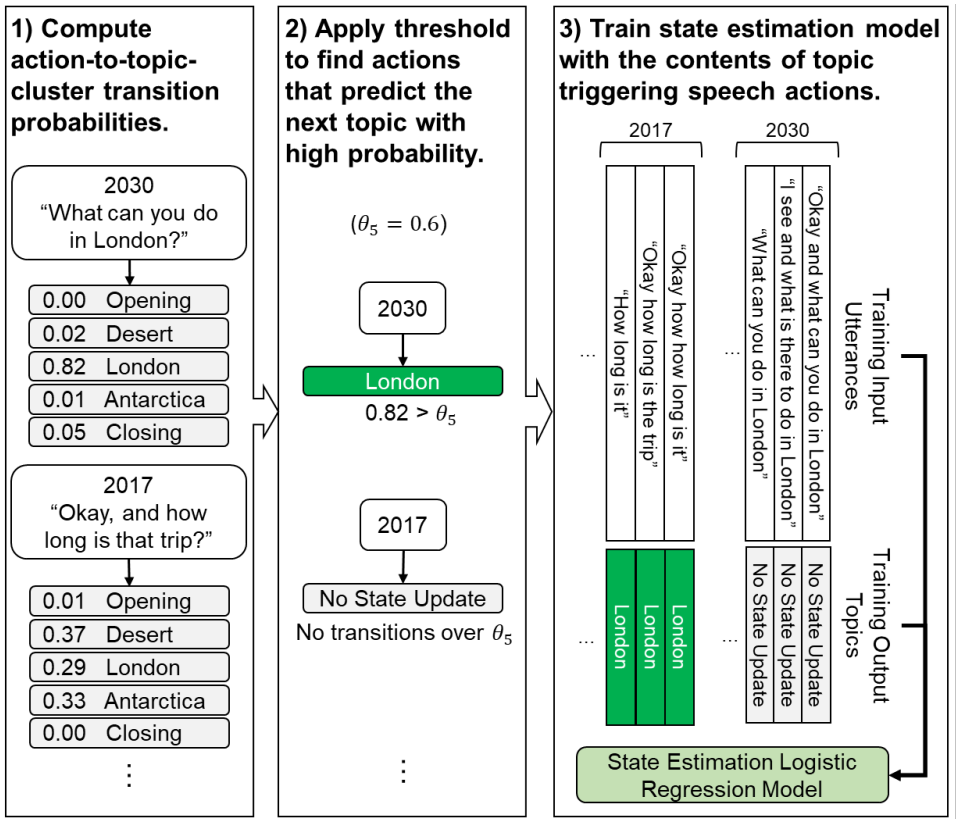


Figure 4.4: Procedure for training the state estimator

the topic clustering algorithm only identifies a subset of the topic-specific actions. To estimate the topic state for robot action prediction a topic state estimator was designed (Fig. 4.2).

The set of possible topic states S was defined to represent the topics of conversation discovered by the topic clustering algorithm. That is, each topic cluster τ, has a corresponding topic state s_τ, $s_\tau \in S$. Each timestep t of an interaction has some topic state s_t. At each step the topic state estimator determines whether the topic state s_t remains the same or changes based on the most recent action a_t.

The topic state estimator is a logistic regression classifier that takes an utterance vectorization as input and outputs the new topic state. The topic state is estimated after each utterance, but since some utterances, such as ambiguous questions, do not change the topic of conversation, an additional class, 'no state update', was used to indicate no update or insufficient evidence to update the topic state. Training the topic state estimator consists of three steps (Fig. 4.4).

1) Compute action-to-topic-cluster transition probabilities. For each speech action a the probability of transitioning to topic τ was computed from the training data by finding each instance of a, a_t, and looking ahead at the topic cluster of the next travel agent action, $a_{TA,t+1}$, using (4.5.3). Since customer actions are often ambiguous, the transition probabilities were based on the topic of the next travel agent action, which

provided more information about the topic of conversation.

$$\mathrm{P}_{\mathrm{trans}}(a,\tau) = \frac{|\{a_t;\, a_t{=}a \wedge a_{TA,t+1}{\in}\tau\}|}{|\{a_t;\, a_t{=}a\}|} \tag{4.5.3}$$

2) Apply a threshold, θ_5 ($= 0.60$), to the action-to-topic-cluster transition probabilities to select actions that predict the next topic with high probability. Such actions indicate that the topic state should be updated based on the most highly predicted topic cluster. For example, action 2030, with the *typical utterance* "What can you do in London?", was followed by actions in topic cluster 3 (London) 82% of the time. This is sufficient evidence to change the topic state to s_3 when the system observes utterances similar to those in action 2030.

Actions for which transition probabilities to all topics were below θ_5 were selected as candidate examples of the 'no state update' class. Since this set of actions was large, candidates for training the classifier were randomly selected until the number of 'no state update' example actions was on par with the number of example actions for the other topic classes. Unselected candidate actions were not used to train the classifier.

The advantage of selecting all actions that predict the next topic with high probability, instead of only actions that predict topic *transitions* (between two topics), is that it enables more robust topic state estimation that can recover from errors. For example, when the estimator misses the topic state update at the first utterance in a topic run, there is a chance of updating the topic state correctly during the next utterance in the topic run.

3) Train the logistic regression classifier on the utterances from the customer actions (speech clusters) and topics found in Step 2. The training inputs to the logistic regression classifier consist of the utterance vectorizations of all the utterances in the speech clusters that predict the next topic with high probability. Their corresponding outputs are the most highly predicted topic states s_τ or the 'no state update' class.

Training a logistic regression model on the contents (utterances) of the customer speech actions takes advantage of statistical learning to make the topic state estimation more robust to variations in speech not represented in the training data, while still providing some degree of interpretability since the actions and topics that were used for training are easy to see.

The topic state also needs to be updated based on the robot's actions, which is a simpler procedure than updating based on human actions since the robot's actions are explicitly represented in the system during run-time. Whenever the robot action predictor outputs a speech action that predicts the next topic with high probability, the topic state estimator updates the topic state directly. An example of a robot action that updates the state is, 3016 "Okay, that's a great choice" $\rightarrow$ *Closing*.

After training, the topic state estimator achieved a prediction accuracy of 70.3% on a manually annotated subset of 15 interactions containing 424 utterances.

Input Customer Action	Customer Action Typical Utterance	State	Predicted Travel Agent Action
2001	Hello.	Opening	Hi there, how can I help you today?
2017	Okay, and how long is that trip?	Desert	That is 14 nights.
		London	Five nights.
		Antarctica	That is 10 days.
2022	Okay, I think I'm going to go with the London trip.	Closing	Okay, that's a great choice.

Table 4.5: Examples of rules learned by the system with topic state

4.5.4 Robot Action Prediction with Topic State

After the topic states are estimated for each turn in the training interactions, the interaction rules are trained. The topic state was incorporated into the previously proposed robot action predictor (Fig. 4.2) by including topic state in the conditional probability that is used to learn the interaction rules, as shown in (4.5.4), where s_t is the topic state at time t.

$$a_{TA,t} = \underset{a \in A_{TA}}{\operatorname{argmax}} P\left(a | a_{C,t}, s_t\right) \tag{4.5.4}$$

Thus, a customer speech action / topic state pair is input to the action predictor and the most frequent travel agent action observed following that pair in the training data is returned. The rest of the system works the same as described in Sec. 4.4.

4.5.5 Interaction Rules with Topic State

Readability of the interaction rules learned by the system with topic state (Table 4.5) allows a human designer to understand how the robot will respond to input customer actions, and the rules that include topic state reveal how the topic of conversation affects the robot's decision.

The rules learned with topic state allow the system to respond correctly to ambiguous customer actions. For example, Table 4.5 shows three rules learned for responding to the ambiguous customer action 2017 ("Okay, and how long is the trip?"). Each of the three rules provides the correct information about the travel package indicated by the topic state. Incorporation of topic state solves the problem of ambiguity illustrated in Sec. 4.4.3, where the without-state system could learn only a single response to customer action 2017.

In a second preliminary offline evaluation, the system with topic state responded to ambiguous customer utterances with a higher rate of correctness than the without-state system (62% vs. 39%, $\chi 2(1, N=190) = 17.70$, $p < .001$), and performed better overall on all customer utterances (67% vs. 55%, $\chi 2(1, N=470) = 12.06$, $p < .001$).

Thus, incorporating knowledge of the interaction structure aided in resolving ambiguous speech.

4.6 Offline System Evaluation

A more thorough offline evaluation was conducted on the human-human data to compare the performance of the proposed system to several systems using state-of-the-art techniques.

4.6.1 Experimental Design

The focus of this study was to develop a technique for generating appropriate robot behaviors by modeling the structure of the interaction, so the experimental design compared the behaviors of six action prediction conditions.

Nearest Neighbor worked by matching the customer's utterance to the closest customer utterance in the training data using the cosine distance between the utterance vectors (Sec. 4.3.5). The robot then performed the same travel agent utterance that followed in the training data. This is a commonly used baseline for data-driven dialog systems [100].

Without-state was the same as the proposed system but without the topic state estimation module (presented in Sec. 4.5). This condition was chosen to examine the effects of incorporating the state estimation module.

NMF Topic was the same as the proposed system but used non-negative matrix factorization (NMF) instead of the proposed topic state estimator. NMF, an unsupervised topic modeling algorithm similar to LDA, is a state-of-the-art method for topic modeling of small datasets [4, 136]. The topic model was trained by setting the NMF parameters to discover five topics from the human-human dataset, using the term frequency-inverse document frequency representations of utterances as input. During runtime, the topic state was updated whenever the maximum NMF topic of an utterance was greater than an empirically-set threshold (0.05), otherwise the previous topic was retained. This condition was chosen to compare to an existing topic modeling method.

RNN (recurrent neural network) consisted of a many-to-one recurrent neural network with three layers: an input layer corresponding to the customer utterance vector, a 100 unit RNN layer, and a softmax output layer with each unit corresponding to a robot speech cluster ID. The RNN was trained for 1000 epochs using the categorical cross entropy loss function and the Adagrad update function [24]. The learning rate was 0.001 and the batch size was 128. For prediction, the customer's utterance vectorizations were input and the *typical utterance* of the predicted speech cluster was the final output. RNNs are a widely used machine learning approach for end-to-end dialog systems [83, 128] and other natural language processing tasks [45, 133]; therefore, they are a suitable for comparison.

LSTM (long short term memory) was identical to the RNN system, except the RNN

layer was replaced by an LSTM layer. LSTMs were designed to overcome the shortcomings of RNNs in modeling long term dependencies [49] and have recently been used for dialog systems [83]. The LSTM system can potentially retain topic information in memory for longer, so it was hypothesized to perform better than the RNN system.

Proposed consisted of the complete system introduced in the previous sections.

The prediction accuracies for the subsets of *ambiguous* and *unambiguous* customer actions were also separately analyzed. It was expected that all the systems would perform equally well on the *unambiguous* actions, but the *proposed* system would perform better on *ambiguous* actions than the other systems.

4.6.2 Evaluation Procedure

Hold-one-out cross validation was used to train and test the system. That is, to evaluate on each of the 192 interactions, the system was trained on the other 191 interactions (192 folds). 36 interactions (about a third of the dataset), containing 500 utterance predictions, were randomly selected for evaluation by a human evaluator. An expert annotator labeled the 500 customer utterances as either *ambiguous* or *unambiguous*. The ambiguous subset of the data contained 201 utterance pairs and the unambiguous subset contained 299 utterance pairs.

One human evaluator (F, age 38), blind to the experimental conditions, evaluated each predicted robot action with a binary label of either *correct* or *incorrect*. To receive a *correct* label the robot's action must have provided the correct information. If the customer did not request any specific information, then the criterion was that the robot's action must be socially appropriate. The evaluations were made based on transcripts of the human-human interactions automatically transcribed using ASR. In the case that the customer's utterance was clipped or contained too many ASR errors for the evaluator to determine if the predicted action was correct or not, the instance was not included in the evaluation (6% of all instances). Evaluations by a second evaluator on ten percent of the 500 instances showed a high degree of agreement, with a kappa coefficient of 0.80, so the evaluations were judged to be reliable.

4.6.3 Offline Evaluation Results

Table 4.6 shows the results of the offline evaluation. A Yates' chi-squared test was used to test for statistical significance between the proposed system and the other systems (Table 4.7). "

On the *ambiguous* customer utterances the *proposed* system performed significantly better than all other conditions. On the *unambiguous* customer utterances there were no significant differences between the *proposed* system and the other systems, except the *nearest neighbor* system (*proposed* 74.5% vs. *nearest neighbor 58*.5%, $\chi 2(1, \text{N}=469) = 14.47$, $p < .001$). On *all utterances* the *proposed* system performed significantly better

	Customer Utterances					
	Ambiguous		Unambiguous		All	
Nearest Neighbor	36%	***	59%	***	49%	***
Without-state	38%	***	71%	n.s.	58%	***
NMF Topic	47%	**	73%	n.s.	62%	*
RNN	45%	**	73%	n.s.	62%	*
LSTM	48%	**	**78%**	n.s.	66%	n.s.
Proposed	**62%**		75%		**69%**	

Table 4.6: Offline action prediction results on all customer utterances, and the ambiguous and unambiguous subsets of customer utterances (asterisks represent comparison with *Proposed* * $p<.05$, ** $p<.01$, *** $p<.001$)

		Customer Utterances		
		Ambiguous N=195	Unambiguous N=274	All N=469
Proposed vs. Nearest Neighbor	χ^2	24.63	14.47	37.41
	p	**<.001**	**<.001**	**<.001**
Proposed vs. Without-state	χ^2	20.77	0.59	13.40
	p	**<.001**	.442	**<.001**
Proposed vs. NMF Topic	χ^2	8.11	0.15	5.15
	p	**.004**	.698	**.023**
Proposed vs. RNN	χ^2	10.56	0.04	5.78
	p	**.001**	.846	**.016**
Proposed vs. LSTM	χ^2	7.55	1.01	1.24
	p	**.006**	.314	.265

Table 4.7: Offline evaluation Yate's chi-squared test results

than the other systems, except the *LSTM* (*proposed* 69.3% vs. *LSTM* 65.5%, $\chi 2(1, N=469) = 1.24$, $p = .265$).

The proposed system outperformed all other systems in handling ambiguous customer questions, the focus of this study. Additionally, the proposed system provided a human readable representation of the interaction logic. Although there was no significant difference between the *proposed* system and *LSTM* on *all utterances*, the *proposed* system performed significantly better on the *ambiguous* customer utterances (*proposed* 62.1% vs. *LSTM* 47.7%, $\chi 2(1, N=195) = 7.55$, $p = .006$).

Although LSTMs were designed for modeling long term dependencies, the *LSTM* system failed to learn the topic dependence of ambiguous utterances in this evaluation. LSTMs (and RNNs) typically require very large amounts of data to effectively capture such dependencies, so the small dataset size may have caused their poor performance. The ability to learn from a small dataset is one strength of the proposed system.

Thus, the proposed system outperformed the state-of-the-art unsupervised learning techniques at dealing with ambiguity in human speech in the travel agent interaction

scenario.

4.7 User Evaluation

A user study was conducted to evaluate the proposed system with a real robot interacting with real people. An example trial, and some special cases are shown in the supplementary video.

4.7.1 Experimental Design

The experimental design consisted of a within-participants design with three experimental conditions: (4.4.1) the proposed system, (4.5.1) the without-state system, and (4.5.2) the nearest neighbor based system, as described in Sec. 4.6.1.

4.7.2 Participants

15 external participants (9 male, 6 female, mean age 34.3, s.d. 8.1) were recruited, all fluent English speakers with little to no experience interacting with robots, and no knowledge of the details of the experiment beyond the instruction's contents.

4.7.3 Experimental Setup

Participants interacted with *ERICA*, an android in the form of a young woman (Fig. 4.1) [33]. The android has 19 DOF, including 13 actuators in the face for eye gaze, facial expressions, and speech movements. She uses Hoya's VoiceText software[3] to synthesize speech and ASR available in the Google Speech API to transcribe human speech collected through a handheld microphone. An array of ceiling-mounted sensors allows the robot to track the positions of people in the room and direct her gaze at them [11].

During the interactions, the participant's speech was collected in real time and fed into the run-time system (Fig. 4.2). The robot's actions were selected by the robot action predictor and synthesized using the android's speech synthesis. Facial expressions and gestures (smile, head nod, etc.) were randomized during each robot action to make her more animated. The robot's lip and trunk movements were automatically generated in synchrony with the speaking rhythm based on the audio output from the speech synthesizer [106]. Interactions were recorded using three 1080p high resolution webcams and a microphone for the participant's speech, and the android's speech stream was recorded directly to file.

[3] http://voicetext.jp/

4.7.4 Procedure

The participant played the role of a customer and the robot acted as a travel agent. Participants were instructed to role-play three interactions in each of the experimental conditions (9 interactions per participant). In order to test the robot's behaviors in response to a variety of different customer behavior patterns, each interaction was role-played as one of the three customer types used in the human-human data collection.

Each interaction started with the participant standing near the door, then they were instructed to walk up to and greet the android before stating what they were looking for. Furthermore, they were instructed to finish each interaction by stating their decision – which travel package they wished to purchase or if they were still undecided – and then walk back to the starting location. Lastly, the participants were instructed to treat each interaction as if it was the first and to forget whatever information was collected in previous interactions.

The order of the experimental conditions was varied for each participant and the order of the customer types was kept the same for each condition to reduce ordering effects. After each round of three interactions in one condition the participant was instructed to fill out a questionnaire. Since it was not clear to the participants whether some of the robot's behaviors were correct or not (e.g. when providing a price) the experimenter kept a tally of the number of correct and incorrect robot actions during each condition and provided the participant with that information. The participant then completed a questionnaire.

4.7.5 Measurement

The effects of the experimental conditions on the robot's behavior were measured in two ways. A questionnaire was used to collect the participant's subjective, qualitative judgements of the robot's behaviors, and transcripts of the experiment interactions were annotated to obtain an objective count of the number of correct and incorrect actions.

Questionnaire A questionnaire containing five questions was designed to compare the participants' subjective ratings.

- **Q1. How understandable was the wording and flow of the robot's speech?** (1 = "very difficult to understand", 7 = "very easy to understand"). The *proposed* and *with-state* systems should be the easiest to understand since the robot speaks the *typical utterances* of speech clusters.

- **Q2. Were you able to get the information you asked for?** (1 = "not at all", 7 = "yes, completely"). The *proposed* system should provide the most information since it is designed to respond to ambiguous questions.

- **Q3. How much effort was required to get the information you asked for?** (1 = "completely effortless", 7 = "maximum effort"). The *proposed* system should

require the least effort since it should make fewer mistakes, minimizing the need to rephrase questions.

- **Q4. How well did the robot respond to ambiguous questions?** (1 = "failed completely", 7 = "greatly succeeded"). The *proposed* system should respond to ambiguous questions the most accurately since it can track the topic of conversation.
- **Q5. How do you think the robot performed overall (including the above points)?** (1 = "failed completely", 7 = "greatly succeeded"). The *proposed* system should perform the best overall for the reasons stated above.

Quantitative Measurement In addition to asking the participants their subjective judgements, a quantitative evaluation was conducted in which the robot's actions were rated by a human evaluator.

One evaluator (male, age 22), blind to the experimental conditions, evaluated the robot responses in all 135 interactions (15 participants $\times$ 3 conditions $\times$ 3 interactions per condition) using the same procedure as in Sec. 4.6.1. 3% of all instances were excluded from the evaluation because of ASR errors in the experiment transcript that made them impossible to judge. Evaluations by a second evaluator on ten percent of the data showed a high degree of agreement, with a kappa coefficient of 0.80. So, the evaluations were judged to be reliable.

It was hypothesized that the *proposed* system would have the highest rate of correct actions, followed by the *without-state* system, and the *nearest neighbor* system performing the worst.

4.7.6 Results

Questionnaire Results Table 4.8 shows the results for the questionnaire. One-way repeated measures ANOVA with Greenhouse-Geisser correction determined that there was a statistically significant effect of the experimental conditions on each of the questionnaire responses. Q1. $F(1.283, 17.963) = 14.884$, $p < .001$; Q2. $F(1.642, 22.988) = 14.512$, $p < .001$; Q3. $F(1.554, 21.753) = 19.409$, $p < .001$; Q4. $F(1.746, 24.450) = 18.984$, $p < .001$; and Q5. $F(1.523, 21.322) = 41.323$, $p < .001$.

Post-hoc Tukey tests (Table 4.8) found significant differences between the *proposed* system and the other two systems on all questions, except for the *proposed* and *without-state* systems on Q1.

In conclusion, the questionnaire responses supported the hypotheses, with *proposed* rated the best on all questions, followed by *without-state* (with the exception of Q1), and the *nearest neighbor* system coming in with the worst ratings.

Quantitative Results The last row of Table 4.8 shows the quantitative evaluation results, each percentage representing the mean robot action prediction correctness computed over all experiment trials (N=15) in the indicated experimental condition.

	Nearest Neighbor		Without-state		Proposed
Q1. Understandability	4.2	$p<.001$	**6.0**	$p=.845$	5.8
Q2. Information	3.6	$p<.01$	4.3	$p<.05$	**5.1**
Q3. Effort	4.2	$p<.001$	3.2	$p<.05$	**2.4**
Q4. Ambiguity	2.9	$p<.001$	3.2	$p<.001$	**4.1**
Q5. Overall	3.3	$p<.001$	4.1	$p<.001$	**5.1**
Mean Rate of Correct Robot Actions	49%	$p<.001$	54%	$p<.01$	**65%**

Table 4.8: User study results (p values represent post-hoc Tukey test comparison with *Proposed*

The *proposed* system achieved a mean robot action prediction correctness rate of 65.2%, *without-state* 54.2%, and *nearest neighbor* 49.2%. A one-way repeated measures ANOVA determined that the experimental conditions had a statistically significant effect on the mean rates of robot action correctness: $F(2, 28) = 12.821$, $p < .001$. A post hoc Tukey test (Table 4.8) found statistically significant differences between the *proposed* condition and the two other conditions. Thus, the *proposed* system performed better than the *without-state* and *nearest neighbor* systems. Therefore, this evaluation validates the effectiveness of the proposed system.

4.7.7 Topic State Estimation Results

The proposed topic state estimator estimated the state correctly 69% of the time and recovered on the next turn 18% of the time in the case of errors. Thus, the estimator was able to generalize from the training data and was robust to some errors.

4.7.8 Analysis of Errors

The system made incorrect responses for five reasons: action matching errors (33% of all errors), customer utterances not present in the training data (23%), topic state estimation errors (18%), incorrectly learned rules (18%), and ASR errors (8%). This suggests that the system's performance could be increased by focusing improvements on the utterance-to-action matching mechanism and dealing with out-of-scope customer utterances.

4.8 Discussion and Conclusion

The main objectives of this work fit under the general goal of learning interaction behaviors for a robot using data-driven methods, without input from a human designer. Firstly, the system was to learn how to respond to ambiguous human actions that de-

pend on hidden state. Secondly, the learned behaviors were to be represented in a human-readable way.

4.8.1 Human-Readable Interaction Rules

Human-readability enables debugging of faulty robot behaviors by a designer. In future work a hybrid system could learn social robot interaction logic from data but also allow for manual debugging and adjustment. Additionally, rule interpretability allows a designer to examine the list of robot interaction rules in order to validate them for safe operation.

The proposed system achieves human-readability in two ways: First, by finding an explicit set of interaction rules, which map a clearly-defined set of human actions to a clearly-defined set of robot responses. Second, by defining a discrete set of topics, which makes the dependence on hidden state explicit. Both are important for interpretability of the interaction logic.

In contrast, neural network architectures such as RNNs, LSTMs, and deep neural networks with attention such as presented in [78] may achieve higher prediction accuracy with larger datasets, but they are black boxes. The downside of such machine learning models in general is that the reasoning behind their decisions is relatively unclear to human understanding.

4.8.2 Learning Topic from Action Co-occurrences

This work presents a novel approach to discovering topic by clustering actions based on action co-occurrences. The proposed topic clustering algorithm is unique since it is based on abstract, discrete symbols representing *actions*, and not any lower level information, like words or language. There are at least two advantages to discovering topics through action co-occurrences rather than textual analysis. First, by forgoing textual analysis, and representing actions with discrete symbols instead, the proposed technique can be applied to other languages and even other modalities. Second, techniques that rely on textual analysis (e.g.) are reliant on having error-free, grammatically correct sentences, so it is difficult to apply them to automatically transcribed, natural, human conversation data with many disfluencies and ASR errors.

An additional result of operating at the level of actions is that the topic clustering algorithm learns structure in the pragmatic, social level, rather than the semantic level. This enables it to discover that two actions are about the same topic even when it is not clear from the textual content. For example, the question “How much is the Antarctica vacation?” and the answer “It is $3000” have no distinguishing words that hint they belong to the same topic, but the high frequency of co-occurrence of these two actions suggests that they are about the same thing.

4.8.3 Applicability to Multiple Modalities

Many techniques focus on text, but the proposed system can potentially be applied to any modality whose data can be clustered to discover a set of abstract, discrete actions, such as individuals' walking trajectories and stopping points [78, 79]. In the future this work could be extended to learn facial expressions, hand gestures, and vocal inflections for heightened robot expressivity. Also, visual attention [71] could be integrated to enable communication about the physical world.

4.8.4 Limitations of the Human-human Interaction Dataset

The main limitation of the travel agent dataset (Sec. 4.3) was the small number of 6 participants. In the future our proposed system may be applied to data collected in the real world. Therefore, the lab-collected dataset should have the same characteristics (but at a smaller scale).

It is reasonable to have only a few participants playing the travel agent in the training data, as this reflects the reality of the target domain, where the number of human employees available for training the system is also limited. Moreover, sometimes it may even be desirable to train the system on a single travel agent's data to learn their specific character and interaction style. Therefore, the training dataset is sufficient in this regard.

In contrast, it is important to have variation in customer behavior in the training dataset to demonstrate that the proposed system can learn correct customer speech clusters and interaction rules despite variation.

Greater variation of customer behavior may make customer speech clustering more difficult. However, the core customer actions (asking about travel package features, etc.) would remain the same regardless of the number of customers, so the increase in variation could be offset by collecting more data, containing more examples of the core actions. Thus, while the number of customer participants in the training dataset might have mixed impact on system performance, it is not a critical weakness, and it does not invalidate the study's results.

Furthermore, the number of customers did not affect the proposed system's generalizability: In the offline experiment, in which the system was evaluated on interactions from the training dataset with 6 participants, the rate of correct robot actions was 69%. In the user study, which was conducted with 15 new participants, the rate of correct robot actions was 65%. The small difference in performance between these two experiments (4%) suggests that the proposed system was able to generalize despite the small training dataset size.

4.8.5 Generalizability to broader domains and larger datasets

In a broader domain (e.g. a real, non-limited travel agent scenario with dozens of travel packages) the number of actions and topics would increase. However, with a large enough dataset, with sufficient examples of each action and topic for speech and topic

clustering, the proposed system will still be capable of learning the repeatable aspects of the interaction.

Concerning dataset size (quantity), the set of common abstract actions will remain the same regardless. The number of phrasing variations of actions may increase with the size of the dataset, but so also will the number of examples of each phrasing. The purpose of the proposed system is to learn these common actions, i.e. the repeatable, core aspects of the interaction. Therefore, the proposed system's performance is expected to improve if the size of the dataset increased and the set of abstract actions remained constant.

4.8.6 Conclusion

This work presented a technique for automatically learning social robot interaction behaviors, in the form of human-readable rules, from unannotated human-human interaction examples that are full of ASR errors, natural speech variation, and disfluencies. Additionally, a novel topic clustering algorithm was introduced that discovered topics and phases of interaction based on action co-occurrences, to resolve ambiguous human speech. In evaluation, the proposed system performed significantly better than five other systems. The proposed technique was demonstrated in a travel agency scenario in which human participants interacted with a real robot. In the future this behavior learning approach could be extended to multiple modalities, such as gestures and facial expressions. It would also be beneficial to incorporate online learning to learn during real-time human-robot interaction.

Chapter 5

Conclusion and Future Directions

5.1 Conclusion

The contributions of this thesis are three approaches to modeling *time-persistent interaction context* that solve three problems that limited the previous state-of-the-art approaches in data-driven imitation learning of human social behaviors. In the first study, I showed how a recurrent neural network architecture with gated recurrent units could be used to learn a time persistent **memory representation of important events** in the interaction history, such that the robot could perform appropriate memory-dependent actions. In the second study, I showed how a time-persistent **customer behavior model**, in combination with on-line learning and a curiosity mechanism, could enable a social robot to adapt to individual human differences and generate more varied, interesting behaviors. Finally, in the third study, I proposed a novel approach for learning to model time-persistent **topics of conversation** which enabled a social robot to resolve ambiguous customer questions.

5.2 Future Directions

There are several exciting possible directions for future research on robot imitation learning of human social behavior.

5.2.1 Multi-party interaction

So far, the research on imitation learning of human social behavior has largely been focused on learning behaviors for a robot to interact with one human at a time. But, in the real-wold, there are many scenarios where the robot must interact with multiple humans at once, i.e. multiparty interaction. It remains an open question whether the previous approaches will generalize to interactions with multiple humans, such as a shopkeeper robot interacting with multiple customers or a tour guide robot interacting with a group of tourists, but it seems that they may not. Some questions raised are, how

can the robot learn to distinguish between speech directed at the robot vs. inter-human speech? What new spatial formation models are needed for the robot to appropriately position itself in relation to a group of humans? And, what kind of machine learning architectures are necessary to consider inputs from multiple humans?

5.2.2 Integration with back-end knowledge bases

Nowadays, there are many structured knowledge bases of general information, such as DBpedia - a knowledge base extracted from Wikipedia [73]. Additionally, places like camera shops, where social robots may be useful, typically use inventory management systems that contain information on the shop's products. So far, the data-driven imitation learning approach has been applied purely to data from human-human example interactions. But, in the future it would be useful to leverage databases like those described above in a hybrid data-driven / knowledge-based approach to improve social robots' abilities. For example, perhaps it may be possible to learn relations between the content of human's speech and knowledge base entities to enable a robot to discuss topics from Wikipedia. Moreover, perhaps learning relations between customer / shopkeeper speech and the products in an inventory management system may enable a shopkeeper robot to stay up-to-date on the store's continuously changing product inventory without having to be re-trained.

5.2.3 Application to interactions in the wild

The end goal of this work is to apply the data-driven technique to real-world scenarios. For this, a sensor array would have to be setup in the wild in a place with frequent human-human interactions, for the purpose of collecting training data and providing input to the robot after deployment. Furthermore, a robot behavior learning system would have to be applied to the collected training data.

There are many challenges to such a proposal. Firstly, there may be legal hurdles concerning the privacy of people in public whose data is recorded for training purposes. Secondly, there are the technical challenges regarding the sensor network - an array of human tracking sensors and microphones would have to be deployed, and the collected data would have to be processed to deal with the noise present in a highly populated environment. Moreover, sensor fusion is necessary to map the audio and human tracking data to individuals in the environment.

Then, there is the learning problem to consider. Data collected in the wild will be much messier than the data collected in the lab or simulated data. People could be using various languages that must be distinguished, and their actions may not always be repeatable, which is a challenge for the data-driven imitation learning approach. These challenges provide many opportunities for future research that will bring us closer to the end goal of deploying social robots in the wild.

Acknowledgments

Working on my PhD research in Japan has been an interesting and challenging experience, which I would not have been able to complete on my own. I would like to thank Dr. Hiroshi Ishiguro for taking me on as a PhD student and providing an excellent lab environment and resources with which to conduct my research. I would like to thank Dr. Dylan F. Glas, who supervised the first portion of my research, for his enthusiasm and challenging discussions, which aided me in building a strong research foundation, and Dr. Takayuki Kanda, who supervised the last portion of my research, for his advice and for providing me opportunities to become more involved in the HRI community.

I spent the majority of my time as a PhD student working as a research intern at ATR, and I would like to offer my special thanks to all the people who helped me during that time. I wold like to thank Dr. Takashi Minato, Dr. Carlos T. Ishi, Dr. Satoru Satake, Dr. Kanako Tomita, and the other researchers at ATR for helping me conduct my research; former fellow students Dr. Phoebe Liu and Dr. Deneth Karunarathe for their aid in navigating the PhD process; and Xiqian Zheng and Dr. Soheil Keshmiri for their interesting discussions. I would like to recognize the interns Nick Mulligan, Jhon Orjuela, Carol Fu, and Cio Ellorin for their aid in data collection, and the ATR staff, especially Minaka Homma, Megumi Taniguchi, Chiyo Ochiai, Maki Kiiro, Chiaki Kimura, Yukiko Imamura, Hisae Yasui, Thomas Kaczmarek, and Florent Fierri for their technical and administerial support.

I would also like to thank Makiko Kashioka, Hanako Nakajima, and the other staff at Osaka University for their administerial assistance, Dr. Dana Kulić for her feedback during journal writing, and Dr. Koh Hosoda and Dr. Takayuki Nagai for contributing their time as readers for my thesis.

Finally, I would like to thank my friends and family, especially my parents and Eri Nishikawa for their moral support, and the Nishikawa family for helping with general life in Japan.

This work was supported by JST, ERATO, ISHIGURO symbiotic Human-Robot Interaction Project, Grant Number JPMJER1401 and by the Ministry of Education, Culture, Sports, Science and Technology (MEXT).

Related Publications

International Journal Publications

1. Malcolm Doering, Dylan F. Glas, and Hiroshi Ishiguro, "Modeling Interaction Structure for Robot Imitation Learning of Human Social Behavior", *IEEE Transactions on Human-Machine Systems* (2019).

2. Malcolm Doering, Phoebe Liu, Dylan F. Glas, Dana Kulić, Takayuki Kanda, and Hiroshi Ishiguro, "Curiosity did not kill the robot: A curiosity-based learning system for a shopkeeper robot", *ACM Transactions on Human-Robot Interaction* (in press).

3. Malcolm Doering, Takayuki Kanda, and Hiroshi Ishiguro, "Neural network based memory for a social robot: Learning a memory model of human behavior from data", *ACM Transactions on Human-Robot Interaction* (in press).

International Conference Videos

1. Dylan F. Glas, Malcolm Doering, Phoebe Liu, Takayuki Kanda, and Hiroshi Ishiguro, "Robot's Delight: A Lyrical Exposition on Learning by Imitation from Human-human Interaction", *ACM/IEEE 12th Annual Conference on Human-Robot Interaction (HRI 2017)*, Vienna, Austria, March 2017.
   Best Video Award

Symposium Posters and Workshop Papers

1. Malcolm Doering, Dylan F. Glas, and Hiroshi Ishiguro, "Modeling Interaction Structure for Robot Imitation Learning of Human Social Behavior", *1st International Symposium on Systems Intelligence Division*, Osaka, Japan, January 2018.

2. Phoebe Liu, Malcolm Doering, Dylan F. Glas, Takayuki Kanda, Dana Kulić, and Hiroshi Ishiguro, "Personalizing crowdsourced human-robot interaction through curiosity-driven learning", *ACM/IEEE 14th Annual Conference on Human-Robot Interaction (HRI 2019)*, Daegu, South Korea, March 2019.

Works Cited

[1] H. Admoni and B. Scassellati. Data-driven model of nonverbal behavior for socially assistive human-robot interactions. In *ICMI*, pages 196–199. ACM, 2014.

[2] C. C. Aggarwal and C. Zhai. *A survey of text clustering algorithms*, pages 77–128. Springer, 2012.

[3] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[4] S. Arora, R. Ge, Y. Halpern, D. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu. A practical algorithm for topic modeling with provable guarantees. In *ICML*, pages 280–288, 2013.

[5] P. Baxter and G. Trafton. Cognitive architectures for human-robot interaction. In *2014 9th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 504–505. IEEE, 2014.

[6] D. M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[7] K. E. Boyer, R. Phillips, E. Y. Ha, M. D. Wallis, M. A. Vouk, and J. C. Lester. Modeling dialogue structure with adjacency pair analysis and hidden markov models. In *NAACL HLT*, pages 49–52. ACL, 2009.

[8] K. E. Boyer, R. Phillips, A. Ingram, E. Y. Ha, M. Wallis, M. Vouk, and J. Lester. Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden markov modeling approach. *IJAIED*, 21(1-2):65–81, 2011.

[9] C. Breazeal, N. DePalma, J. Orkin, S. Chernova, and M. Jung. Crowdsourcing human-robot interaction: New methods and system evaluation in a public environment. *JHRI*, 2(1):82–111, 2013.

[10] A. Breuing and I. Wachsmuth. Let's talk topically with artificial agents! providing agents with humanlike topic awareness in everyday dialog situations. In *ICAART*, volume 2, 2012.

[11] D. Brscic, T. Kanda, T. Ikeda, and T. Miyashita. Person tracking in large public spaces using 3-d range sensors. *IEEE THMS*, 43(6):522–534, 2013.

[12] H. Bunt. Context representation for dialogue management. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 77–90. Springer, 1999.

[13] M. Cakmak and A. L. Thomaz. Designing robot learners that ask good questions. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 17–24. ACM, 2012.

[14] M. T. K. Chan, R. Gorbet, P. Beesley, and D. Kulić. Curiosity-based learning algorithm for distributed interactive sculptural systems. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 3435–3441. IEEE, 2015.

[15] C.-W. Chang, J.-H. Lee, P.-Y. Chao, C.-Y. Wang, and G.-D. Chen. Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school. *Educational Technology & Society*, 13(2):13–24, 2010.

[16] H. Chen, X. Liu, D. Yin, and J. Tang. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35, 2017.

[17] S. Chernova, N. DePalma, E. Morant, and C. Breazeal. Crowdsourcing human-robot interaction: Application from virtual to physical worlds. In *RO-MAN, 2011 IEEE*, pages 21–26. IEEE, 2011.

[18] S. Chernova, J. Orkin, and C. Breazeal. Crowdsourcing hri through online multiplayer games. In *AAAI Fall Symposium: Dialog with Robots*, pages 14–19, 2010.

[19] K. Cho, B. Van MerriÃ«nboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[20] K. Clark and C. D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *EMNLP*, 2016.

[21] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 1996.

[22] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[23] M. Doering, D. F. Glas, and H. Ishiguro. Modeling interaction structure for robot imitation learning of human social behavior. *IEEE Transactions on Human-Machine Systems*, 49(3):219–231, June 2019.

[24] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[25] A. Duranti and C. Goodwin. *Rethinking context: Language as an interactive phenomenon*, volume 11. Cambridge University Press, 1992.

[26] L. El Asri, H. Schulz, S. Sharma, J. Zumer, J. Harris, E. Fine, R. Mehrotra, and K. Suleman. Frames: a corpus for adding memory to goal-oriented dialogue systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 207–219, 2017.

[27] A. Ezen-Can and K. E. Boyer. Understanding student language: An unsupervised dialogue act classification approach. *JEDM*, 7(1):51–78, 2015.

[28] G. Fischer. User modeling in human-computer interaction. *User modeling and user-adapted interaction*, 11(1-2):65–86, 2001.

[29] M. E. Foster, S. Keizer, Z. Wang, and O. Lemon. Machine learning of social states and skills for multi-party human-robot interaction. In *Proceedings of the workshop on Machine Learning for Interactive Systems (MLIS 2012)*, page 9, 2012.

[30] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

[31] D. Glas, S. Satake, T. Kanda, and N. Hagita. An interaction design framework for social robots. In *RSS*, volume 7, page 89, 2012.

[32] D. F. Glas, T. Kanda, and H. Ishiguro. Human-robot interaction design using interaction composer: Eight years of lessons learned. In *HRI*, pages 303–310. IEEE Press, 2016.

[33] D. F. Glas, T. Minato, C. T. Ishi, T. Kawahara, and H. Ishiguro. Erica: The erato intelligent conversational android. In *RO-MAN*, pages 22–29. IEEE, 2016.

[34] R. Gockley, A. Bruce, J. Forlizzi, M. Michalowski, A. Mundell, S. Rosenthal, B. Sellner, R. Simmons, K. Snipes, and A. C. Schultz. Designing robots for long-term social interaction. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1338–1343. IEEE, 2005.

[35] E. Goffman. *Frame analysis: An essay on the organization of experience.* Harvard University Press, 1974.

[36] G. Gordon, C. Breazeal, and S. Engel. Can children catch curiosity from a social robot? In *HRI*, pages 91–98. ACM/IEEE, 2015.

[37] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[38] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204, 1986.

[39] B. J. Grosz, S. Weinstein, and A. K. Joshi. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225, 1995.

[40] D. Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2017.

[41] S. Guo, J. Lenchner, J. H. Connell, M. Dholakia, and H. Muta. Conversational bootstrapping and other tricks of a concierge robot. In *HRI*, pages 73–81, 2017.

[42] E. T. Hall. *The hidden dimension*, volume 609. Garden City, NY: Doubleday, 1966.

[43] M. Henderson. Machine learning for dialog state tracking: A review. In *MLSLP*, 2015.

[44] M. Henderson, B. Thomson, and S. Young. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *SLT*, pages 360–365. IEEE, 2014.

[45] M. Henderson, B. Thomson, and S. Young. Word-based dialog state tracking with recurrent neural networks. In *SIGDIAL 15*, pages 292–299, 2014.

[46] T. Hester and P. Stone. Intrinsically motivated model learning for developing curious robots. *Artificial Intelligence*, 247:170–186, 2017.

[47] R. Higashinaka, N. Kawamae, K. Sadamitsu, Y. Minami, T. Meguro, K. Dohsaka, and H. Inagaki. Unsupervised clustering of utterances using nonparametric bayesian methods. In *InterSpeech*. ISCA, 2011.

[48] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[49] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[50] L. Hong and B. D. Davison. Empirical study of topic modeling in twitter. In *SOMA*, pages 80–88. ACM, 2010.

[51] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[52] S. Janarthanam and O. Lemon. Adaptive generation in dialogue systems using dynamic user modeling. *Computational Linguistics*, 40(4):883–920, 2014.

[53] K. Jokinen, H. Tanaka, and A. Yokoo. Context management with topics for spoken dialogue systems. In *COLING*, pages 631–637. ACL, 1998.

[54] D. Jurafsky and J. H. Martin. *Speech and language processing*, volume 3. Pearson London, 2014.

[55] J. Kahn, Peter H., J. H. Ruckert, T. Kanda, H. Ishiguro, A. Reichert, H. Gary, and S. Shen. Psychological intimacy with robots?: Using interaction patterns to uncover depth of relation. In *Proc. HRI*, pages 123–124. IEEE Press, 2010.

[56] T. Kanda, T. Hirano, D. Eaton, and H. Ishiguro. Interactive robots as social partners and peer tutors for children: A field trial. *Human-Computer Interaction*, 19(1):61–84, 2004.

[57] F. Kaplan and P.-Y. Oudeyer. From hardware and software to kernels and envelopes: a concept shift for robotics, developmental psychology, and brain sciences. Report, Cambridge University Press, 2011.

[58] H. Kawai, T. Toda, J. Ni, M. Tsuzaki, and K. Tokuda. Ximera: A new tts from atr based on corpus-based technologies. In *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[59] Y. Kim, J. Bang, J. Choi, S. Ryu, S. Koo, and G. G. Lee. Acquisition and use of long-term memory for personalized dialog systems. In *International Workshop on Multimodal Analyses Enabling Artificial Agents in Human-Machine Interaction*, pages 78–87. Springer, 2014.

[60] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[61] T. Kitade, S. Satake, T. Kanda, and M. Imai. Understanding suitable locations for waiting. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pages 57–64. IEEE Press, 2013.

[62] A. Kobsa. User modeling in dialog systems: Potentials and hazards. *AI & society*, 4(3):214–231, 1990.

[63] H. Kozima, M. P. Michalowski, and C. Nakagawa. Keepon. *International Journal of Social Robotics*, 1(1):3–18, 2009.

[64] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pages 1378–1387, 2016.

[65] K. Kuwamura, S. Nishio, and H. Ishiguro. *Designing Robots for Positive Communication with Senior Citizens*, pages 955–964. Springer, 2016.

[66] K. Kuwamura, S. Nishio, and S. Sato. Can we talk through a robot as if face-to-face? long-term fieldwork using teleoperated robot for seniors with alzheimer's disease. *Frontiers in Psychology*, 7, 2016.

[67] J. E. Laird. *The Soar cognitive architecture*. MIT Press, 2012.

[68] T. K. Landauer, P. W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[69] R. Langevin. Is curiosity a unitary construct? *Canadian Journal of Psychology/Revue canadienne de psychologie*, 25(4):360, 1971.

[70] P. Langfelder, B. Zhang, and S. Horvath. Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for r. *Bioinformatics*, 24(5):719–720, 2008.

[71] P. Lanillos, J. F. Ferreira, and J. Dias. Designing an artificial attention system for social robots. In *IROS*. IEEE, 2015.

[72] E. Law, V. Cai, Q. F. Liu, S. Sasy, J. Goh, A. Blidaru, and D. Kulić. A wizard-of-oz study of curiosity in human-robot interaction. In *RO-MAN*, 2017.

[73] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[74] I. Leite, C. Martinho, and A. Paiva. Social robots for long-term interaction: a survey. *International Journal of Social Robotics*, 5(2):291–308, 2013.

[75] I. Leite, A. Pereira, A. Funkhouser, B. Li, and J. F. Lehman. Semi-situated learning of verbal and nonverbal content for repeated human-robot interaction. In *ICMI*, pages 13–20. ACM, 2016.

[76] Z. Li, D. He, F. Tian, W. Chen, T. Qin, L. Wang, and T.-Y. Liu. Towards binary-valued gates for robust lstm training. In *ICML*, 2018.

[77] Z. C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[78] P. Liu, D. F. Glas, T. Kanda, and H. Ishiguro. Data-driven hri: Learning social behaviors by example from human-human interaction. *IEEE T-RO*, 32(4):988–1008, 2016.

[79] P. Liu, D. F. Glas, T. Kanda, and H. Ishiguro. Learning proactive behavior for interactive social robots. *Autonomous Robots*, pages 1–19, 2017.

[80] P. Liu, D. F. Glas, T. Kanda, and H. Ishiguro. Two demonstrators are better than one-a social robot that learns to imitate people with different interaction styles. *IEEE Transactions on Cognitive and Developmental Systems*, 2017.

[81] P. Liu, D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. How to train your robot-teaching service robots to reproduce human social behavior. In *RO-MAN*, pages 961–968. IEEE, 2014.

[82] Y. Liu, K. Han, Z. Tan, and Y. Lei. Using context information for dialog act classification in dnn framework. In *EMNLP*, pages 2160–2168, 2017.

[83] R. T. Lowe, N. Pow, I. V. Serban, L. Charlin, C.-W. Liu, and J. Pineau. Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue & Discourse*, 8(1):31–65, 2017.

[84] J. F. Maas, T. Spexard, J. Fritsch, B. Wrede, and G. Sagerer. Biron, what's the topic? a multi-modal topic tracker for improved human-robot interaction. In *RO-MAN*, pages 26–32. IEEE, 2006.

[85] K. Madani, C. Sabourin, and D. M. RamÃk. *Artificial Curiosity Emerging Human-Like Behavior: Toward Fully Autonomous Cognitive Robots*, pages 501–516. Springer, 2016.

[86] D. Martens, B. Baesens, T. Van Gestel, and J. Vanthienen. Comprehensible credit scoring models using rule extraction from support vector machines. *EJOR*, 183(3):1466–1476, 2007.

[87] M. P. Michalowski, S. Sabanovic, and H. Kozima. A dancing robot for rhythmic social interaction. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 89–96. IEEE, 2007.

[88] T. D. Midgley, S. Harrison, and C. MacNish. Empirical verification of adjacency pairs using dialogue segmentation. In *SIGdial*, pages 104–108. ACL, 2009.

[89] S. Müller, S. Sprenger, and H.-M. Gross. Online adaptation of dialog strategies based on probabilistic planning. In *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on*, pages 692–697. IEEE, 2014.

[90] K. P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. Thesis, University of California, Berkeley, 2002.

[91] Y. Nagai, C. Muhl, and K. J. Rohlfing. Toward designing a robot that learns actions from parental demonstrations. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3545–3550. IEEE, 2008.

[92] J. Orkin and D. Roy. The restaurant game: Learning social behavior and language from thousands of players online. *Journal of Game Development*, 3(1):39–60, 2007.

[93] J. Orkin and D. Roy. Automatic learning and generation of social behavior from collective human gameplay. In *AAMAS*, pages 385–392. IFAAMAS, 2009.

[94] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286, 2007.

[95] J. Perez, Y. L. Boureau, and A. Bordes. Dialog system & technology challenge 6 overview of track 1-end-to-end goal-oriented dialog learning. In *DSTC6 Dialog System Technology Challenges*, 2017.

[96] R. P. A. Petrick and M. E. Foster. What would you like to drink? recognising and planning with social states in a robot bartender domain. *Proceedings of CogRob*, 2012, 2012.

[97] E. Pot, J. Monceaux, R. Gelin, and B. Maisonnier. Choregraphe: a graphical tool for humanoid robot programming. In *RO-MAN*, pages 46–51. IEEE, 2009.

[98] M. Purver, T. L. Griffiths, K. P. KÃ¶rding, and J. B. Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. In *COLING*, pages 17–24. ACL, 2006.

[99] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa, and H. Ishiguro. Intrinsically motivated reinforcement learning for human-robot interaction in the real-world. *Neural Networks*, 2018.

[100] A. Ritter, C. Cherry, and W. B. Dolan. Data-driven response generation in social media. In *EMNLP*, pages 583–593. ACL, 2011.

[101] R. Rojas. *The backpropagation algorithm*, pages 149–182. Springer, 1996.

[102] S. Rosenthal, J. Biswas, and M. Veloso. An effective personal mobile robot agent through symbiotic human-robot interaction. In *AAMAS*, pages 915–922. IFAAMAS, 2010.

[103] B. C. Ross. Mutual information between discrete and continuous data sets. *PLoS ONE*, 9(2):e87357, 2014.

[104] C. A. Rothkopf and D. H. Ballard. Credit assignment in multiple goal embodied visuomotor behavior. *frontiers in Psychology*, 1, 2010.

[105] H. Sacks, E. A. Schegloff, and G. Jefferson. A simplest systematics for the organization of turn-taking for conversation. *Language*, pages 696–735, 1974.

[106] K. Sakai, T. Minato, C. T. Ishi, and H. Ishiguro. Speech driven trunk motion generating system based on physical constraint. In *RO-MAN*, pages 232–239. IEEE, 2016.

[107] H. A. Samani, A. D. Cheok, F. W. Ngiap, A. Nagpal, and M. Qiu. Towards a formulation of love in human-robot interaction. In *RO-MAN*, pages 94–99. IEEE, 2010.

[108] J. M. Santos, T. KrajnÃk, and T. Duckett. Spatio-temporal exploration strategies for long-term autonomy of mobile robots. *Robotics and Autonomous Systems*, 88:116–126, 2017.

[109] S. E. Schaeffer. Graph clustering. *Computer science review*, 1(1):27–64, 2007.

[110] J. Schmidhuber. *Maximizing fun by creating data with easily reducible subjective complexity*, pages 95–128. Springer, 2013.

[111] H. Schulz, J. Zumer, L. E. Asri, and S. Sharma. A frame tracking model for memory-enhanced dialogue systems. *arXiv preprint arXiv:1706.01690*, 2017.

[112] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, 2016.

[113] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[114] K. Severinson-Eklundh, A. Green, and H. HÃ$\frac{1}{4}$ttenrauch. Social and collaborative aspects of interaction with a service robot. *Robotics and Autonomous systems*, 42(3):223–234, 2003.

[115] C. Shi, T. Kanda, M. Shimada, F. Yamaoka, H. Ishiguro, and N. Hagita. Easy development of communicative behaviors in social robots. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5302–5309, 2010.

[116] M. Shiomi, D. Sakamoto, T. Kanda, C. T. Ishi, H. Ishiguro, and N. Hagita. A semi-autonomous communication robot-a field trial at a train station. In *HRI*, pages 303–310. IEEE, 2008.

[117] T. Sinha, Z. Bai, and J. Cassell. Curious minds wonder alike: Studying multimodal behavioral dynamics to design social scaffolding of curiosity. *arXiv preprint arXiv:1705.00204*, 2017.

[118] L. Song. The role of context in discourse analysis. *Journal of Language Teaching and Research*, 1(6):876, 2010.

[119] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT*, 2015.

[120] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Dialogue*, 26(3), 2006.

[121] S. Sukhbaatar, J. Weston, and R. Fergus. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[122] J. Thomason, A. Padmakumar, J. Sinapov, J. Hart, P. Stone, and R. J. Mooney. Opportunistic active learning for grounding natural language descriptions. In *Conference on Robot Learning*, pages 67–76, 2017.

[123] A. L. Thomaz and C. Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *AAAI*, volume 6, pages 1000–1005, 2006.

[124] Z. Tian, R. Yan, L. Mou, Y. Song, Y. Feng, and D. Zhao. How to make context more useful? an empirical study on context-aware neural conversational models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–236, 2017.

[125] R. Toris, D. Kent, and S. Chernova. The robot management system: A framework for conducting human-robot interaction studies through crowdsourcing. *Journal of Human-Robot Interaction*, 3(2):25–49, 2014.

[126] G. Trafton, L. Hiatt, A. Harrison, F. Tamborello, S. Khemlani, and A. Schultz. Act-r/e: An embodied cognitive architecture for human-robot interaction. *Journal of Human-Robot Interaction*, 2(1):30–55, 2013.

[127] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, and M. Fiore. Spencer: A socially aware service robot for passenger guidance and help in busy airports. In *Field and Service Robotics*, pages 607–622. Springer, 2016.

[128] O. Vinyals and Q. Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[129] W. Wahlster and A. Kobsa. *User models in dialog systems*, pages 4–34. Springer, 1989.

[130] H. M. Wallach. Topic modeling: beyond bag-of-words. In *ICML*, pages 977–984. ACM, 2006.

[131] Q. A. Wang. Probability distribution and entropy as a measure of uncertainty. *Journal of Physics A: Mathematical and Theoretical*, 41(6):065004, 2008.

[132] J. Weston, S. Chopra, and A. Bordes. Memory networks. In *ICRL*, 2015.

[133] J. Williams, A. Raux, and M. Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33, 2016.

[134] J. D. Williams, A. Raux, D. Ramachandran, and A. W. Black. The dialog state tracking challenge. In *SIGdial*, pages 404–413, 2013.

[135] F. Yamaoka, T. Kanda, H. Ishiguro, and N. Hagita. How close?: model of proximity control for information-presenting robots. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 137–144. ACM, 2008.

[136] X. Yan, J. Guo, S. Liu, X. Cheng, and Y. Wang. Learning topics in short texts by non-negative matrix factorization on term correlation matrix. In *SDM*, pages 749–757. SIAM, 2013.

[137] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM, 1996.