A Socially-Aware Conversational Recommender System for Personalized Recipe Recommendations

Florian Pecune
University of Glasgow
florian.pecune@glasgow.ac.uk

Lucile Callebert
University of Glasgow
lucile.callebert@glasgow.ac.uk

Stacy Marsella
University of Glasgow
stacy.marsella@glasgow.ac.uk

ABSTRACT

One potential solution to help people change their eating behavior is to develop conversational systems able to recommend healthy recipes. Beyond the intrinsic quality of the recommendations themselves, various factors might also influence users' perception of a recommendation. Two of these factors are the conversational skills of the system and users' interaction modality. In this paper, we present Cora, a conversational system that recommends recipes aligned with its users' eating habits and current preferences. Users can interact with Cora in two different ways. They can select predefined answers by clicking on buttons to talk to Cora or write text in natural language. On the other hand, Cora can engage users through a social dialogue, or go straight to the point. We conduct an experiment to evaluate the impact of Cora's conversational skills and users' interaction mode on users' perception and intention to cook the recommended recipes. Our results show that a conversational recommendation system that engages its users through a rapport-building dialogue improves users' perception of the interaction as well as their perception of the system.

KEYWORDS

conversational recommender system; healthcare; socially-aware

ACM Reference Format:
Florian Pecune, Lucile Callebert, and Stacy Marsella. 2020. A Socially-Aware Conversational Recommender System for Personalized Recipe Recommendations. In *Proceedings of the 8th International Conference on Human-Agent Interaction (HAI '20), November 10–13, 2020, Virtual Event, NSW, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3406499.3415079

1 INTRODUCTION

Healthy eating implies complex decision making processes [6], including being aware of healthy options and choosing among them [24]. One solution to overcome this issue and help people to make healthier choices is to develop health-aware food recommender systems [31]. While significant effort has been put recently into optimizing the food selection algorithms [30], many other factors can also influence users' overall experience when interacting with a recommender system [14]. Indeed, the way the recommendation is presented [18], the system's response time [33], or even the length of the system's utterances [20] can have an influence on users' perception of the system.

One trend to improve users' experience is to make the interaction more natural by designing the recommendation process as a conversation [23]. Besides helping users to achieve task-oriented goals, conversations can also fulfill interpersonal functions, such as building rapport [29]. Rapport can be described as a dynamic process that can be achieved when people "click" with each other or feel the interaction is due to "chemistry" [27]. Human-human studies have found that rapport between two people can influence task performance in situations as diverse as peer-tutoring [25] and negotiation [7]. Based on these findings, it becomes important to endow recommender systems with social conversational infrastructure that would allow them to build rapport with their users to improve task effectiveness.

In this paper, we present a conversational system able to recommend recipes matching users' needs while building rapport with them. More specifically, our work focuses on investigating how the conversational skills of a recipe recommender system and the interaction modes it offers to its users would influence users' perception and their intention to cook. First, we describe the design of our system and its architecture before we explain how the recommendation process works. Then, we evaluate our system through an experiment in which we study the impact of our system's conversational skills and interaction mode on its persuasiveness. Our main contributions are (1) a rapport-building conversational approach to deliver recipe recommendations adapted to users' needs and habits and (2) a subjective evaluation investigating the influence of a recommender system's conversational skills and interaction mode on users' perception of the system, users' perception of the interaction and users' intention to cook the recommended recipes.

2 RELATED WORK

Food Recommender Systems. A common approach for food recommender systems is to recommend a recipe based on its ingredients. In [8], for example, the authors developed a system that relies on recipes that people like to infer their preferred ingredients. The system then recommends new recipes containing the previously inferred ingredients. In [9], the authors developed a system that collects users' preferences by asking them to rate and tag the recipes they usually cook at home. The system then relies on user's preferences to rank recipes and deliver recommendations with the highest scores. This Matrix Factorization algorithm outperformed the content-based approach proposed by [8]. Other approaches only rely on dietary information to recommend recipes that would match users' needs. YumMe, the recommender system developed in [36], automatically extracts dietary information from pictures of recipes to form a user profile. The system then relies on this user

HAI '20, November 10–13, 2020, Virtual Event, NSW, Australia

ACM ISBN 978-1-4503-8054-6/20/11…$15.00
https://doi.org/10.1145/3406499.3415079

profile to deliver subsequent recommendations. In [11], authors analyzed people's eating behavior and clustered people in two categories: those interested in getting healthy recipes, and those who did not care about that. They found that two of the main recipe rating predictors for the first group were the fat and calorific content of the recipe, and decided to incorporate these features in their recommendation process.

All these works focus on improving recommendation algorithms. They do not investigate how the modality of the interaction between the system and its users can improve users' experience which, according to [15], should not be neglected.

Conversational Recommender Systems As pointed out by [23], one way to improve users' experience during complex search settings is to endow recommender systems with conversational skills. One of the first attempts comes from [35], where the authors analyze a corpus of interactions between human recommenders and users to derive two main recommendation phases: Interview and Delivery. The interview phase consists in a sequence of questions which purpose is to gather relevant preferences from the user. The goal of the delivery phase is to actually deliver the recommendation – or a list of recommendations – based on the information gathered during the interview phase. After presenting the user with a recommendation, the system must be prepared to respond to the user's reactions, or critiques [4]. For instance, the user may want to change his preferences, specify whether he thinks an item is interesting or not – in which case the system must update the user's preferences accordingly –, or if they would like to see more similar items [28]. More recently, approaches such as [37] proposed to optimize these two phases by teaching their conversational recommender system which questions to ask users to deliver the most accurate recommendation. Rather than focusing on system's dialogue policy or strategies, some other works try to find the best way for users to interact with a recommender system. The authors in [19] investigate different interaction modes by objectively comparing three versions of a same music recommender systems. The first one used buttons to interact with users, the second one used free text, and the last one was a hybrid version that used buttons whenever there was a need to disambiguate. Results show that the hybrid version had a better interaction cost and recommendation accuracy than the two other versions.

All these systems aim at fulfilling the task goals of a conversation. However, this is not the only goal that people want to achieve during an interaction [29] and interpersonal goals such as building and maintaining a good relationship or rapport should also be considered when designing conversational systems.

Rapport-Building Conversational Systems Researchers have already started to investigate rapport-building conversational systems in different contexts, and study how rapport-building strategies influence agents' task-performance. With Rea the virtual Real Estate Agent, authors investigated how small-talk influenced the price users were ready to invest in a new house [1]. Closer to our current work, some researchers specifically focused on rapport-building conversational agents in the context of a recommendation task. The authors of [16] evaluated the impact of self-disclosures and reciprocity on a conversational recommender system's perceived performance. The results showed that both self-disclosures and reciprocity had a significant positive impact on users' satisfaction with the interaction, and intention to use the system. However, their system was not fully autonomous (they used a Wizard of Oz) and they did not try to measure the impact of a system's self-disclosures on the perceived quality of the recommendation. Recently, [21] presented a conversational recommender system able to draw from the various explanations humans use with one another. The authors demonstrated that users preferred movie recommendations coming from a system that was able to justify its choice using its "own" personal opinion and talk about its "own" personal experience related to the recommended movie.

Although all these works rely on rapport-building conversational strategies, few of them investigate how rapport-building dialogues influence the perceived quality of the items recommended, or people's compliance towards the recommendations. Moreover, they do not investigate the impact of users' interaction mode on users' perceptions. In this paper, we aim at building a conversational recommender system that recommends recipes while building rapport with its users. More specifically, in this paper, we focus on the following research questions:

RQ1: How does the way users interact with a conversational recommender system influence their perception of and their intention to cook recommended recipes?

RQ2: How do a conversational recommender system's conversational strategies influence users' perception of and their intention to cook recommended recipes?

3 CONVERSATIONAL AGENT DESIGN

To answer to our research questions, we built and deployed Cora, a conversational agent that recommends recipes to its users through a rapport-building dialogue. We explain below how we designed our system, describe its architecture (see Fig.1) and detail its recommendation process.

3.1 System Design

Works in grounded cognition have shown that people's eating behaviors are driven by many mechanisms that they are most often not aware of [3]. For example, different foods can be associated to different emotions. Each food can then be described in terms of the emotion it elicits in people, and assigned a global emotional score. Similarly, foods can be described in terms of *healthiness* and *fillingness* (how full does one feel after eating that food?) and assigned fillingness and healthiness scores. As these two dimensions have been found to be good predictors for the frequency to which a food is eaten and for the acceptance rate of food recommender systems [11], we rely on them to design our recommendation process. Cora uses on a knowledge-based approach to recommend recipes to its users. Cora first establishes a user profile based on users' eating habits and needs. Cora then selects an ingredient matching this profile, and recommends recipes using all this information.

3.1.1 Task-Oriented Dialogue. To design our task-oriented dialogue, we conducted a pilot involving approximately 100 participants. This pilot helped us to identify the different questions that Cora would have to ask before actually recommending a recipe.

The interaction scenario is designed to follow the traditional interview/delivery structure [14, 35]. During the interview phase, Cora first greets the user and introduce itself before it starts gathering the relevant pieces of information it needs to deliver personalized recipe recommendations. Cora asks whether the user feels hungry, whether they want to eat healthy, and whether they are on a specific diet. Cora also asks whether the user wants to use a specific ingredient for their recipe. Finally, Cora asks how much time the user is willing to spend to cook dinner. After that question, Cora enters the delivery phase and recommends a recipe. From then, the user can accept or reject the recommendation. In both cases, the system updates its knowledge about the user's preferences and asks whether the user would like another recipe. If the user declines, Cora says goodbye and the interaction ends. The user can also ask for another recipe, which results in another recommendation from the system. If the user refines their preferences, (e.g. by saying the don't like one of the ingredients) the system updates its knowledge accordingly and recommends another recipe.

3.1.2 Rapport-Building Dialogue. For Cora to build rapport with its users, we mostly rely on the computational model exposed in [38] and its list of conversational strategies. More specifically, we implemented the following strategies:

Small Talk as an introductory phase. Small talk usually consists of safe and non-intimate questions to break the ice during a first interaction with someone [26]. Previous work in the domain of conversational agents emphasized the role of small talk in task oriented contexts [1]. In our work, small-talk consists of Cora asking four questions at the beginning of the interaction before reaching the preference gathering phase. Cora first asks about users' name and whether they are doing alright. Then, Cora asks about what the users usually eat for dinner, and why they eat such food.

Self-Disclosures. Disclosing personal information about yourself during an interaction has been linked to affiliative interpersonal outcomes such as liking and trust [17]. Previous work already emphasized the need to endow conversational systems with the ability to self-disclose personal information [16]. Cora thus discloses information about itself to its users during the small-talk and preference gathering phases (e.g. "I try to eat healthy dinners myself" or "I love to spend time cooking").

Feedback and Acknowledgements. Acknowledgements are a way to show understanding of a previous utterance during a conversation [32]. In this work, Cora uses such acknowledgements (e.g. "okay", "right", "sure", etc...) to show that it understood what the user just said. Cora also uses reciprocal appreciation to give a feedback to what users said and build rapport with them at the same time. As explained by [12], people tend to appreciate their interlocutor more when they express similar attitudes toward an opinion, an object, or another person. For instance, if one user says that he/she is hungry, Cora gives a feedback saying "I'm hungry too!"

Personal opinions as explanations. As shown in [21], a system justifying its recommendation using its "own" personal opinion or personal experience can increase rapport during an interaction. In this work, Cora uses explanations such as "It's personally one of my favorites!" or "This recipe is delicious." whenever it recommends a recipe to its user.

3.2 Architecture

3.2.1 Front-End. The front-end consists of a web page that allows each client to communicate with the server. Chat messages are displayed in a single scroll-down chat window. Whenever Cora recommends a recipe, a poster including the recipe's picture, ingredients, and cooking steps is displayed in the conversation. The user initiates the conversation by saying "Hello Cora". To have better control over turn taking, the user cannot send a second message to Cora until the first one was answered. We set up two different interfaces (**User-Mode**) for users to chat with Cora. In the *chat-mode*, the users write their messages to Cora by typing free text in a text input at the bottom of the chat window. In the *buttons-mode*, the users select pre-defined messages to send to Cora by clicking on buttons and/or selecting options in drop-down menus. For example, to the question "What do you think about this recipe?" users can answer either "I like it, thank you!", "No, I don't like the recipe", "No, I don't like #ingredient" or "No, other reason". All the answer options were defined based on the most recurrent answers given by users in our prior pilot.

3.2.2 Back-End. The back-end consists of a server developed in Python which handles multiple simultaneous client connections and disconnections. For each new client, the server creates a dedicated Cora-agent. A Cora-agent is composed of three modules: 1) the Natural Language Understanding (NLU) module, in charge of making sense of what the user is saying, 2) a Dialog Manager (DM), deciding what to say next based on the output of the NLU and 3) a Natural Language Generation (NLG) module, generating sentences in natural language based on the output of the DM. Each module is described in more details below. The server is then in charge of distributing clients' messages to the corresponding Cora-agents as well as Cora-agents' messages to the corresponding clients.

Natural Language Understanding. The first component triggered is the Natural Language Understanding (NLU) module, which extracts communicative intentions and entities from users' utterances. Our NLU module uses the Python libraries nltk and Spacy to do lemmatization, dependency parsing and POS tagging on the utterance. We extract relevant entities by matching words/lemmas with a set of entities (i.e. ingredients, diets and intolerances) provided by the Spoonacular API[1] that we use to recommend recipes. Given the entities, the POS tags and the dependency tags, the NLU module then uses a set of rules to determine the user's intent, the associated entity and entity-type as well as a valence. For instance, the output corresponding to the input "I don't like mushrooms" is `{intent: "inform", entity_type: "food", entity: "mushroom", valence: "-"}`. This information is sent to the DM.

Dialog Management. We designed our Dialog Manager (DM) as a finite state machine. It takes the user intent, entity-type, entity and valence extracted from the user's utterance as inputs; it then uses these to transition to each new state based on the current state of the dialog and a set of rules (see section 3.1 for an overview of

[1] https://spoonacular.com/food-api

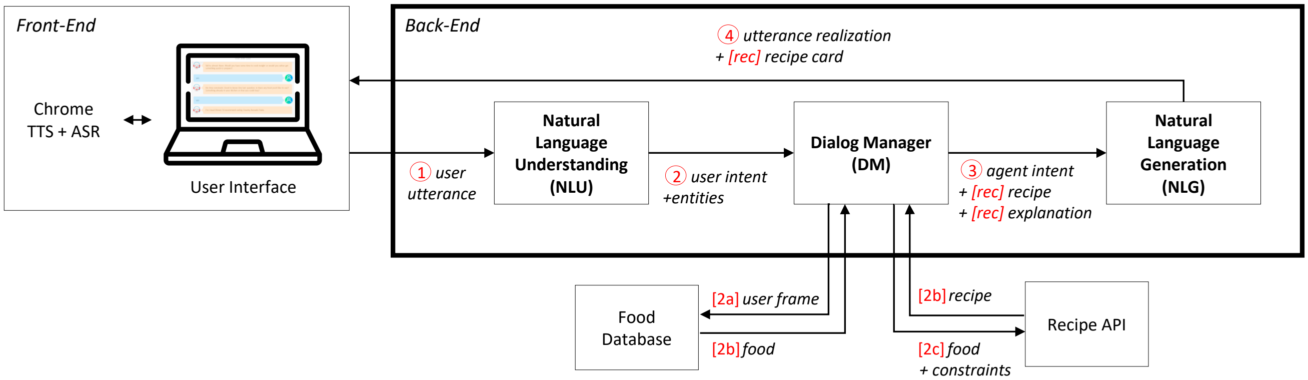

Figure 1: Architecture of Cora, our COnversational Recommender Agent, with recommendation-only items preceded by brackets.

the scenario). The DM stores the user's recognized entities to keep track of their preferences in a *user frame*. The content of the user frame is used during the recommendation process as described in section 3.3.

We defined two interaction modes for Cora (**Cora-Mode**). In the *task-conv* mode, Cora focuses exclusively on its recommendation task. In the *social-conv* mode, Cora uses all the conversational strategies defined in 3.1 to build rapport with its users, including small-talk at the beginning of the conversation. We therefore defined a specific finite state machine for the *social-conv* to include additional states corresponding to the small-talk phase.

Natural Language Generation. Given the user-utterance data outputted by the NLU, the dialog state outputted by the DM and the interaction style (**Cora-Mode**), the Natural Language Generation (NLG) module uses a lookup table to generate a utterance in natural language. In the *social-conv* mode, the generated utterance contains one or more rapport-building strategies, which is not the case in the *task-conv* mode. Figure 2 presents two samples of interactions depicting the difference between the Cora-Modes and User-Modes.

3.3 Recommendation process

Our main challenge for the recommendation process is to deliver a recipe that best matches users preferences. To better understand these preferences, we take into account both users' habits (trait preferences) and users' current desires (state preferences). We first collected a Food database regrouping the 40 ingredients that people most frequently cook and eat for dinner. Each ingredient of the database is characterized by healthiness and fillingness scores in $[-1, 1]$ that were computed by averaging individual healthiness and fillingness scores assigned by hundreds of participants (e.g salmon is associated with a healthiness value of 0.926 and a fillingness value of 0.678).

3.3.1 Trait preferences. The trait preferences are gathered through a questionnaire that users answer prior to the interaction with Cora. Users are asked to rate how frequently they eat seven specific foods using 7-point Likert scales (anchors: 0 = never, 6 = once a day). These seven foods were selected from our healthiness-fillingness food database, after filtering out foods that are not compatible with a vegan diet (e.g. steak). Specifically, we selected the two items of our database with the highest ratings and the two items with the lowest ratings on the healthiness dimension and on the fillingness dimension, thus obtaining two sets of four items each. The two sets of selected food having one item in common, the final set has seven items.

To compute a trait healthiness preference score $p_t(h) \in [-1, 1]$ for a user, we first calculate, for each food j, a healthiness score $s_j(h)$ as:

$$s_j(h) = \begin{cases} freq_j & \text{if } j \text{ is a healthy food} \\ -freq_j & \text{if } j \text{ is not a healthy food,} \end{cases}$$

where $freq_j$ is the frequency at which the user eats food j, as self-reported on our Likert-scale. We then sum those scores to obtain the user's trait healthiness score $p_t(h) = \sum_j s_j(h)$. We proceed in a similar way to calculate a trait fillingness preference score $p_t(f)$.

Even though this data is collected prior to the beginning of interaction, it is stored in the DM's user frame and available during the interaction.

3.3.2 State preferences. The state preferences are collected during the interaction with Cora and correspond to the answers to "How healthy do you want your meal to be?" for healthiness and "How hungry are you?" for fillingness. The NLU extracts from the user's utterance a desired level of healthiness (resp. fillingness) converted to a value $p_s(h)$ (resp. $p_s(f)$) in $\{-1, -.75, -.5 - .25, 0, .25, .5 .. 75, 1\}$. For example, "no preference" will be mapped to a value of 0 while "slightly" will be mapped to a value of .25.

The user's preferences are then averaged over trait and state values, resulting in two values $p(h)$ and $p(f)$ in $[-1, 1]$.

Besides the healthiness and fillingness dimensions, the data collected during our pilot study showed that three other elements are critical when it comes to recommend a recipe: the diet / intolerances of the user (e.g. vegan or intolerant to gluten), the amount of time the user is willing to spend cooking, and whether there is a specific ingredient the user wants to use. Those preferences are also gathered during the interview phase of the dialog with Cora.

3.3.3 Giving a recommendation. When the interview phase is over, Cora recommends recipes to the user. To do so, the DM goes through the following steps: 1) find an ingredient if the user did not specify one during the interview phase, 2) find a recipe with this ingredient and 3) use the user's feedback for subsequent recommendations.

1) Ingredient. If the user did not provide one, we want to select the best ingredient for the user to use in the recipe. Leveraging our Food Database, the DM generates a list of preferred ingredients *ingredients_list* for the user. To do so, the user's preferences for healthiness and fillingness are represented as a vector $p = \langle p(h), p(f) \rangle$. Each ingredient of the Food Database is represented in the same vector space and ingredients are sorted by the distance of their vector to p, with the closest ingredient as the first one of the list. Ingredients that the user dislikes or cannot eat (e.g. if vegan) are excluded from the list.

2) Recipe. To recommend a recipe to the user, the DM uses the Spoonacular API. This API allows us to query the Spoonacular database for recipes including or excluding a specific ingredient, that correspond to a specific diet (e.g. vegan) and/or take into account intolerances (e.g. to gluten) and that can be cooked in a specific amount of time. Each recipe received from the API is described by a title, a list of ingredients, a list of preparation steps (i.e. instructions), and some nutritional information, including a global healthiness score and an amount of calories that we map to our fillingness score. The DM queries the API for two recipes containing the first ingredient of the *ingredients_list* and stores the results in a *recipes_list*. Similarly to ingredients, recipes of the *recipes_list* are represented in the healthiness-fillingness vector space and are sorted by distance to the preference vector p. If the DM receives less than two results from the API or if at any point the *recipes_list* is empty, the first ingredient in *ingredients_list* is popped out of the list and the DM queries the API for two more recipes following the same procedure.

3) Using user's feedback. If Cora has to give more than one recommendation, it uses the user's feedback to select another recipe: i) If the user answered they don't like an ingredient (resp. recipe), the disliked ingredient is stored in a *disliked_ingredients_list* (resp. *disliked_recipes_list*). The DM then removes from *recipes_list* all recipes that contain disliked ingredients, as well as recipes with a title similar to disliked recipes. We do fuzzy matching to compute the distance between two titles, using Levenshtein distance. If *recipes_list* is empty, the DM goes back to step 2.

4 EXPERIMENT

To answer to our research questions **RQ1** and **RQ2**, we designed an experiment investigating how our system's conversational skills and interaction mode influenced the perceived quality of both the system and the interaction, in addition to users' intention to cook recommended recipes.

4.1 Stimuli

For the sake of the experiment, we identified two different independent variables, corresponding to the DM/NLG modes and the front-end interfaces described in section 3.2. The first one represents Cora's conversational mode (**Cora-Mode**) as a between-subject independent variable with two levels: socially-oriented conversation (*social-conv*) in which Cora liven up the conversation by using rapport-building strategies as described in 3.1, and task-oriented conversation (*task-conv*) in which Cora simply asks questions and delivers recommendations. The second between-subject variable represents the way the user can interact with Cora (**User-Mode**) and has two levels: a button mode (*buttons-mode*) in which users interact with Cora using buttons and drop-down lists only, and a chat mode (*chat-mode*) in which users can type whatever they want in natural language.

Our experiment has a 2x2 design with **Cora-Mode** and **User-Mode** as between subject variables. Two samples of interactions depicting the difference between the conditions are presented in figure 2. In each of the four conditions, participants followed the same procedure. They were first presented with a consent form informing them about the conditions of the experiment. Those who agreed to partake in the study were then presented with a short description of the task explaining the context of the interaction (i.e., the participant has to find a recipe to cook for tonight). Before interacting with Cora, participants had to fill in a questionnaire about their eating habits as described in section 3.3.1. Each participant was then randomly assigned to a group according to the different independent variables and interacted with Cora following the scenario described in section 3.1. After the end of their interaction, participants took three surveys that measured the quality of the interaction, the quality of the system and their intention to cook the recipes recommended to them. In addition to these three surveys, two open-ended questions asked participants their thoughts about Cora and about the experiment. Finally, participants answered a demographics questionnaire.

4.2 Measurements

We measured the three following constructs in our experiment. (a) We relied on rapport [27] -a notion commonly used in the domain of human-agent interactions [10, 39]- as a proxy to measure the quality of the interaction. The eight different items we used to measure rapport are listed in Table 1. (b) The perceived quality of the conversational recommender system was measured using a questionnaire derived from [22]. The seven different items we used encompass multiple aspects of a recommender system's task performance and are listed in Table 2. We also added an eighth item to assess the perceived healthiness of the recipes recommended. (c) Finally, we measured participants' intention the cook the recommended recipes through a questionnaire adapted from [5]. The five items we used to measure intention to cook are listed in Table 3. All answers for the three questionnaires were on a 7-point Likert scale (anchors: 0 = completely disagree, 6 = completely agree).

4.3 Hypotheses

We learned from previous work that rapport-building conversational strategies increase users' satisfaction [16, 21] and users' intention to use the system [16]. Hence, we expect to find a similar positive impact on the perceived quality of the system and on user's intention to cook. We hypothesize the following:

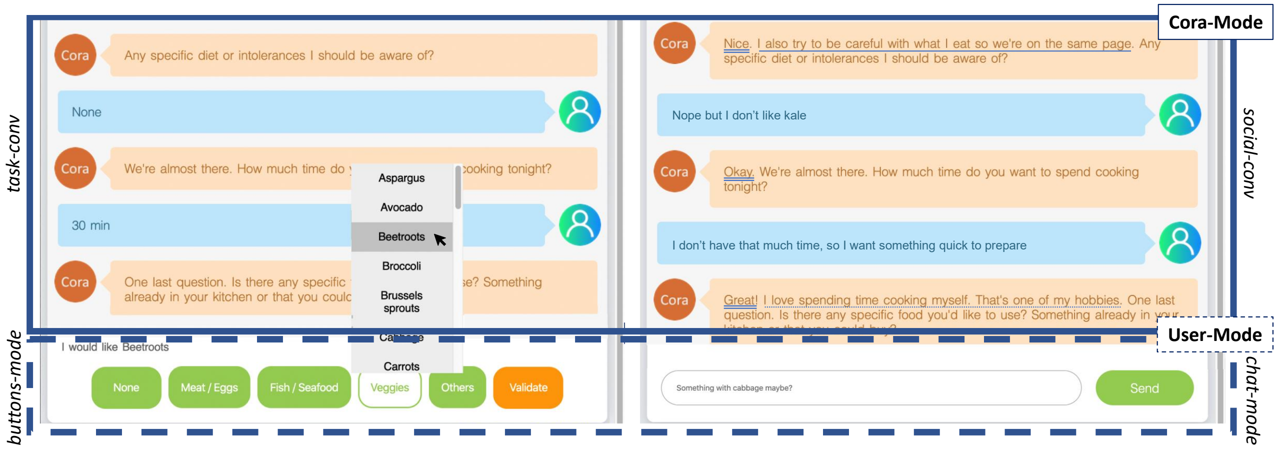

Figure 2: Excerpts of interactions between Cora and its user showing the two Cora-Modes (*task-conv* vs *social-conv*) and the two User-Modes (*buttons-mode* vs *chat-mode*) used in our experiment. The rapport-building conversational strategies used by Cora in the *social-conv* mode are underlined in the dialogue: acknowledgments with a double line, reciprocal appreciations with a single line and self-disclosures with a dotted line.

H1-a: The system's conversational mode (**Cora-Mode**) will have a main effect on the perceived quality of the interaction. More specifically, the interactions with a system that engages participants using rapport-building conversational strategies (*social-conv*) will be perceived as better than the interactions with a system that engages participants through a task-oriented conversation (*task-conv*).

H1-b: The system's conversational mode (**Cora-Mode**) will have a main effect on the perceived quality of the conversational recommender system. More specifically, the quality of a system that engages participants using rapport-building conversational strategies (*social-conv*) will be perceived as higher than the quality of a system that engages participants through a task-oriented conversation (*task-conv*).

H1-c: The system's conversational mode (**Cora-Mode**) will have a main effect on the participants' intention to cook. More specifically, participants interacting with a system engaging them using rapport-building conversational strategies (*social-conv*) will be more likely to report they want to cook one of the recommended recipes compared to participants interacting with a system engaging them through a task-oriented conversation (*task-conv*).

Building rapport is a dyadic process, which would require the user to reciprocate the use of conversational strategies during the interaction [38]. Thus, we expect that allowing users to chat freely with the system will increase their overall experience. We hypothesize the following:

H2-a: Users' interaction mode (**User-Mode**) will have a main effect on the perceived quality of the interaction. More specifically, participants chatting with a system (*chat-mode*) will perceive the interaction as better compared to participants interacting with a system using buttons and drop-down menus (*buttons-mode*).

H2-b: Users' interaction mode (**User-Mode**) will have a main effect on the perceived quality of the conversational recommender system. More specifically, participants chatting with a system (*chat-mode*) will perceive its quality as higher compared to participants interacting with a system using buttons and drop-down menus (*buttons-mode*).

H2-c: Users' interaction mode (**User-Mode**) will have a main effect on the participants' intention to cook. More specifically, participants chatting with a system (*chat-mode*) will be more likely to report they want to cook one of the recommended recipes compared to participants interacting with a system using buttons and drop-down menus (*buttons-mode*).

5 RESULTS

We collected 106 interactions on Amazon Mechanical Turk (N = 106), with a balanced number of interactions per condition. To ensure the quality of the data collected, all participants had at least a 95% approval rate with more than 100 previous HITs validated. Participants were paid USD 0.75 and spent an average of 6 minutes and 11 seconds (std = 3min59s) on the task. Participants were mostly men (67%), from the U.S. (67%), working full-time (80%), with a degree of higher education (89%). They were aged 18-29 (24%), 30-49 (65%) or 50-69 (11%). All participants cook at least occasionally and most of them cook at least once a day (43%) or several times a week (44%). Most participants said they are very familiar (55%) or somewhat familiar (41%) with conversational assistants.

Our conversational agent recommended 238 recipes in total, with an average of 2.24 recommendations per interaction (std = 1.69). The average acceptance rate for the recommendations was 0.79 (std = 0.39).

5.1 Quality of the interaction

We conducted a 2x2 factorial MANOVA (i.e., multivariate analysis of variance) with **Cora-Mode** and **User-Mode** as between-subject factors. The dependent measures were the eight questions presented in Table 1. The factorial MANOVA revealed a significant main effect of **Cora-Mode** ($F(1, 102) = 5.088$; $p < .05$) on the perceived quality of the interaction. There was no main effect of **User-Mode** (F(1,

Dimensions	Subjective items	Cora-Mode		User-Mode	
		task-conv	**social-conv**	**button-mode**	**chat-mode**
Coordination	I felt I was in sync with Cora.	4.58(±1.72)	4.85(±1.07)	4.77(±1.22)	4.66(±1.62)
	I was able to say everything I wanted to say during the interaction.	4.81(±1.57)	5.00(±1.33)	4.77(±1.52)	4.94(±1.39)
Mutual Attentiveness	Cora was interested in what I was saying.	4.36(±1.52)	4.85(±1.25)	4.79(±1.19)	4.42(±1.58)
	Cora was respectful to me and considered to my concerns.	5.21(±1.02)	5.26(±.97)	5.26(±.89)	5.21(±1.09)
Positivity	Cora was warm and caring.	4.21(±1.57)	4.64(±1.30)	4.72(±1.46)	4.13(±1.40)
	Cora was friendly to me.	4.85(±1.22)	5.23(±1.04)	5.17(±.95)	4.91(±1.31)
Rapport	Cora and I established rapport.	3.53(±1.60)***	4.68(±1.30)***	4.23(±1.53)	3.98(±1.60)
	I felt I had no connection with Cora.	2.72(±2.33)	2.11(±2.13)	2.30(±2.21)	2.53(±2.30)

Table 1: Subjective questionnaire adapted from [39] to measure users' perceived quality of the interaction.

Dimensions	Subjective items	Cora-Mode		User-Mode	
		task-conv	**social-conv**	**button-mode**	**chat-mode**
Decision Confidence	The recipes recommended to me during this interaction matched my preferences.	4.58(±1.71)	4.70(±1.21)	4.91(±1.10)	4.38(±1.74)
User Control	Cora allowed me to specify and change my preferences during the interaction.	4.53(±1.56)	4.75(±1.40)	4.98(±1.16)	4.30(±1.69)
Intention to Return	I would use Cora to get recipe recommendations in the future.	4.25(±1.89)*	4.98(±1.14)*	4.87(±1.45)	4.36(±1.71)
Perceived Effort	I easily found the recipes I was looking for.	4.21(±1.94)	4.75(±1.15)	4.74(±1.44)	4.23(±1.72)
Healthiness	The recipes recommended by Cora were healthy.	4.64(±1.30)	5.04(±1.03)	4.94(±1.14)	4.74(±1.23)
Recommendation Quality	I was satisfied with the recipes recommended to me.	4.51(±1.68)	4.94(±1.17)	4.85(±1.22)	4.60(±1.66)
Perceived Usefulness	Cora provided sufficient details about the recipes recommended.	4.09(±1.70)*	4.72(±1.47)*	4.55(±1.43)	4.26(±1.77)
Transparency	Cora explained her reasoning behind the recommendations.	3.34(±1.99)	3.98(±1.86)	3.57(±1.97)	3.75(±1.93)

Table 2: Subjective questionnaire adapted from [22] to measure users' perceived quality of the system.

Dimensions	Subjective items	Cora-Mode		User-Mode	
		task-conv	**social-conv**	**button-mode**	**chat-mode**
Intention to Cook	I want to make the recipe recommended to me.	3.98(±1.72)	4.68(±1.24)	4.57(±1.35)	4.09(±1.67)
	I expect to make the recipe recommended to me.	4.40(±1.71)	4.75(±1.11)	4.77(±1.27)	4.38(±1.59)
	It is likely I will make the recipe recommended to me.	4.28(±1.72)	4.72(±1.03)	4.68(±1.30)	4.32(±1.54)
	I intend to make the recipe recommended to me.	4.11(±1.68)	4.45(±1.27)	4.55(±1.37)	4.02(±1.57)
	I will try to make the recipe recommended to me.	4.15(±1.81)	4.53(±1.34)	4.60(±1.47)	4.08(±1.68)

Table 3: Subjective questionnaire adapted from [5] to measure users' intention to cook.

102) = 1.214; p = .27) on the perceived quality of the interaction and the interaction between the two variables was not significant (F(1; 102) = 3.453; p = .06). H1-a is validated, but not H2-a.

For our follow-up analysis, we looked at univariate effects of **Cora-Mode** on each dependent measure with Student's t-tests. Our results showed a significant main effect of **Cora-Mode** on the item "Cora and I established rapport" (F(1, 104) = 16.24; p < .001; η^2 = .14). This result shows that the version of Cora engaging the participants using conversational strategies was effectively able to build rapport with them. For all the questionnaire items, the system was rated with higher scores when engaging participants in a social dialogue (*social-conv*) compared to a task dialogue (*task-conv*). The quality of the interaction was rated high across all conditions (mean and std for the quality of the interaction across all conditions m = 4.58, std = 1.04). We report a summary of all means and standard errors (in parentheses) for the eight dependent variables in Table 1. The differences between the means are marked according to their level of significance (* for p < .05, ** for p < .005 and *** for p < .001).

5.2 Quality of the system

We conducted a 2x2 factorial MANOVA with **Cora-Mode** and **User-Mode** as between-subject factors. The dependent measures were the eight questions presented in Table 2. The factorial MANOVA revealed a significant main effect of **Cora-Mode** (F(1, 102) = 4.325; p < .05) on the perceived quality of the system. There was no main effect of **User-Mode** (F(1, 102) = 2.354; p = .13) on the perceived quality of the system and the interaction between the two variables was not significant (F(1; 102) = 1.167; p = .28). H1-b is validated, but not H2-b.

Similar to the previous section, we performed a follow-up analysis that looked at univariate effects of **Cora-Mode** on each dependent measure with Student's t-tests. Our results showed a significant main effect of **Cora-Mode** on the participants Intention to Return (F(1, 104) = 5.77; p < .05; η^2 = .05) and Perceived Usefulness (F(1, 104) = 4.00; p < .05; η^2 = .04). Participants are more willing to use a system able to engage them in a rapport-building conversation, and they also perceive that a rapport-building system delivers more details about the recommendations. For all the questionnaire items, the system was rated with higher scores when engaging participants in a social dialogue (*social-conv*) compared to a task dialogue (*task-conv*). The quality of the system was rated high across all conditions (mean and std for the quality of the system across all conditions m = 4.50, std = 1.16). In Table 2, we report a summary of all means and standard errors (in parentheses) for the eight dependent variables. The differences between the means are marked according to their level of significance (* for p < .05, ** for p < .005 and *** for p < .001).

5.3 Intention to cook

We conducted a 2x2 factorial MANOVA with **Cora-Mode** and **User-Mode** as between-subject factors. The dependent measures were the five questions presented in Table 3. The factorial MANOVA revealed no significant main effect of **Cora-Mode** ($F(1, 102) = 2.660$; $p = .1$) or **User-Mode** ($F(1, 102) = 2.852$; $p = .09$) on the perceived quality of the system. The interaction between the two variables was not significant ($F(1; 102) = 2.233$; $p = .14$). H1-c and H2-c are not validated. For all the questionnaire items, the system was rated with higher scores when engaging participants in a social dialogue (*social-conv*) compared to a task dialogue (*task-conv*). The scores obtained by the *button-mode* version were also higher than the scores obtained by the *chat-mode* version for all the items. The intention to cook was rated high across all conditions (mean and std for intention to cook across all conditions m = 4.41, std = 1.39).

5.4 Discussion

The good ratings obtained across each condition combined with the high acceptance rate show that participants have been generally satisfied with Cora and its recommendations. Regardless of the quality of the recommendations, our results also show that endowing recommender systems with rapport-building abilities has a positive influence on users' perception. That is corroborated by the fact that the rapport-building version of Cora systematically obtained better scores than its task-oriented counterpart, and lower standard deviations. In other words, participants preferred the rapport-building version of Cora, and their ratings of this version were more consistent. Furthermore, not only are participants significantly more willing to use a system able to engage them in a rapport-building conversation, but they also perceive that a rapport-building system delivers significantly more details about the recommendations, although it simply gives its "own" personal opinion. The latter result is consistent with [21] and highlights the importance of endowing recommender systems with the ability to express their own opinions about the items they recommend.

Contrary to our initial intuition, we found that the chat version of Cora generally obtained lower scores compared to its button counterpart. However, we noticed two interesting results. First, participants who interacted with a rapport-building Cora via free text rated the quality of the interaction higher than the ones who interacted with the rapport-building Cora via buttons and drop-down menus. This shows that allowing users to freely reciprocate a recommender system's conversational strategies during the interaction helps to improve their overall experience. Participants who interacted with the rapport-building Cora via free text found the system *"fun and engaging"* and thought Cora *"sounded so sympathetic and gave off a vibe of someone who cares about people's opinions"*. Second, the difference in the ratings of the quality of interaction and intention to cook between button-mode and chat-mode was lower when Cora was using rapport-building conversational strategies. In other words, rapport-building strategies mitigated the issues related to natural language understanding. Participants were more forgiving towards the system when it was using rapport-building strategies, as hinted by one comment: *"Cora kind of ignored my dietary preferences, but she sounded pretty natural compared to most chat bots"*.

We found a potential explanation for the lower scores of chat-mode Cora by looking at the interactions logs. Indeed, some specific sentences written by users in the chat-mode were not correctly understood by Cora. We identified two categories of errors: (1) sentences that were misclassified by our natural language understanding components leading to an inaccurate recommendation in the end and (2) users' requests that were not handled by our system, hence not classified at all, leading to users' frustration. Both categories had a negative impact on users' rating of system quality and intention to cook. Furthermore, we found that people who were not familiar with conversational assistants rated the chat-mode version of Cora significantly lower compared to the button-mode. These participants might have been oblivious of the current conversational assistant limitations or conventions and preferred more conventional interfaces that they found more reliable. Although we expected the button-mode to be too limiting, only one participant commented that *"some answers felt restrictive and made me feel like I wouldn't be able to say what I meant with the ready-made answers."*

Finally, one comment from a participant who interacted with rapport-building Cora shed light on a very interesting point: *"It was good, but there's too much unimportant conversation in my opinion. There should be two modes: talkative and time-efficient"*. This is consistent with the work presented in [13] in which the authors classified the users of a recommender system in two categories: those who actively wanted to build rapport with the recommender system, and those who wanted to get a recommendation in the most efficient way. Therefore, building systems able to identify the type of users they are interacting with and to adapt their strategies would consequently improve users' experience.

6 CONCLUSION

In this paper, we presented Cora, a conversational recommender system able to recommend recipes matching users' eating habits and needs. Cora was able to engage its users in a rapport-building dialogue or a task-oriented one, and was able to interact with them using free-text or buttons and drop-down menus. We conducted a user study to evaluate the influence of Cora's conversational skills and users' interaction mode on users' perception and intention to cook. Our results show that endowing conversational recommendation systems with rapport-building conversational strategies significantly improves users' perception of the interaction and of the system itself. We also found that rapport-building strategies were a way to mitigate and lower the impact of the system's misunderstandings on users perception and intention to cook.

One potential extension of this work would be to replace the recipe API we are relying on for recommendations by our own recommendation engine. We are currently considering the use of Knowledge graphs as described in [2, 34] to deliver more personalized recipes recommendation to users. We also want our system to ask a follow-up question whenever users reject a recommendation to understand why they rejected it. That would help the system to refine its users' profiles and to deliver more accurate recommendations later on. We are also interested in introducing a healthiness bias in the recommendation process that would incite users to eat healthier recipes, and evaluate how such a bias would influence users perception.

REFERENCES

[1] Justine Cassell and Timothy Bickmore. 2003. Negotiated collusion: Modeling social language and its relationship effects in intelligent agents. *User modeling and user-adapted interaction* 13, 1-2 (2003), 89–132.

[2] Rose Catherine and William Cohen. 2016. Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 325–332.

[3] Jing Chen, Esther K Papies, and Lawrence W Barsalou. 2016. A core eating network and its modulations underlie diverse eating phenomena. *Brain and cognition* 110 (2016), 20–42.

[4] Li Chen and Pearl Pu. 2009. Interaction design guidelines on critiquing-based recommender systems. *User Modeling and User-Adapted Interaction* 19, 3 (2009), 167.

[5] Mark Conner, Paul Norman, and Russell Bell. 2002. The theory of planned behavior and healthy eating. *Health psychology* 21, 2 (2002), 194.

[6] Sally Jo Cunningham and David Bainbridge. 2013. An analysis of cooking queries: Implications for supporting leisure cooking. (2013).

[7] Aimee L Drolet and Michael W Morris. 2000. Rapport in conflict resolution: Accounting for how face-to-face contact fosters mutual cooperation in mixed-motive conflicts. *Journal of Experimental Social Psychology* 36, 1 (2000), 26–50.

[8] Jill Freyne and Shlomo Berkovsky. 2010. Intelligent food planning: personalized recipe recommendation. In *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 321–324.

[9] Mouzhi Ge, Mehdi Elahi, Ignacio Fernaández-Tobías, Francesco Ricci, and David Massimo. 2015. Using tags and latent factors in a food recommender system. In *Proceedings of the 5th International Conference on Digital Health 2015*. ACM, 105–112.

[10] Jonathan Gratch, Anna Okhmatovskaia, Francois Lamothe, Stacy Marsella, Mathieu Morales, Rick J van der Werf, and Louis-Philippe Morency. 2006. Virtual rapport. In *International Workshop on Intelligent Virtual Agents*. Springer, 14–27.

[11] Morgan Harvey, Bernd Ludwig, and David Elsweiler. 2013. You are what you eat: Learning user tastes for rating prediction. In *International Symposium on String Processing and Information Retrieval*. Springer, 153–164.

[12] Fritz Heider. 2013. *The psychology of interpersonal relations*. Psychology Press.

[13] Alankar Jain, Florian Pecune, Yoichi Matsuyama, and Justine Cassell. 2018. A user simulator architecture for socially-aware conversational agents. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*. 133–140.

[14] Michael Jugovac and Dietmar Jannach. 2017. Interacting with recommenders—overview and research directions. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 7, 3 (2017), 10.

[15] Joseph A Konstan and John Riedl. 2012. Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction* 22, 1-2 (2012), 101–123.

[16] SeoYoung Lee and Junho Choi. 2017. Enhancing user experience with conversational agent for movie recommendation: Effects of self-disclosure and reciprocity. *International Journal of Human-Computer Studies* 103 (2017), 95–105.

[17] Youngme Moon. 2000. Intimate exchanges: Using computers to elicit self-disclosure from consumers. *Journal of consumer research* 26, 4 (2000), 323–339.

[18] Theodora Nanou, George Lekakos, and Konstantinos Fouskas. 2010. The effects of recommendations' presentation on persuasion and satisfaction in a movie recommender system. *Multimedia systems* 16, 4-5 (2010), 219–230.

[19] Fedelucio Narducci, Pierpaolo Basile, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2019. An investigation on the user interaction modes of conversational recommender systems for the music domain. *User Modeling and User-Adapted Interaction* (2019), 1–34.

[20] Florian Pecune, Jingya Chen, Yoichi Matsuyama, and Justine Cassell. 2018. Field Trial Analysis of Socially Aware Robot Assistant. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1241–1249.

[21] Florian Pecune, Shruti Murali, Vivian Tsai, Yoichi Matsuyama, and Justine Cassell. 2019. A Model of Social Explanations for a Conversational Movie Recommendation System. In *Proceedings of the 7th International Conference on Human-Agent Interaction*. ACM, 135–143.

[22] Pearl Pu, Li Chen, and Rong Hu. 2011. A user-centric evaluation framework for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 157–164.

[23] Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*. ACM, 117–126.

[24] Benjamin Scheibehenne, Rainer Greifeneder, and Peter M Todd. 2010. Can there ever be too many options? A meta-analytic review of choice overload. *Journal of consumer research* 37, 3 (2010), 409–425.

[25] Tanmay Sinha and Justine Cassell. 2015. We click, we align, we learn: Impact of influence and convergence processes on student learning and rapport building. In *Proceedings of the 1st Workshop on Modeling INTERPERsonal SynchrONy And infLuence*. ACM, 13–20.

[26] Jan Svennevig. 2000. *Getting acquainted in conversation: A study of initial interactions*. Vol. 64. John Benjamins Publishing.

[27] Linda Tickle-Degnen and Robert Rosenthal. 1990. The nature of rapport and its nonverbal correlates. *Psychological inquiry* 1, 4 (1990), 285–293.

[28] Nava Tintarev and Judith Masthoff. 2007. A survey of explanations in recommender systems. In *2007 IEEE 23rd international conference on data engineering workshop*. IEEE, 801–810.

[29] Karen Tracy and Nikolas Coupland. 1990. Multiple goals in discourse: An overview of issues. *Journal of Language and Social Psychology* 9, 1–2 (1990), 1–13.

[30] Thi Ngoc Trang Tran, Müslüm Atas, Alexander Felfernig, and Martin Stettinger. 2018. An overview of recommender systems in the healthy food domain. *Journal of Intelligent Information Systems* 50, 3 (2018), 501–526.

[31] Christoph Trattner and David Elsweiler. 2017. Food recommender systems: important contributions, challenges and future research directions. *arXiv preprint arXiv:1711.02760* (2017).

[32] David R Traum and Elizabeth A Hinkelman. 1992. Conversation acts in task-oriented spoken dialogue. *Computational intelligence* 8, 3 (1992), 575–599.

[33] Vivian Tsai, Timo Baumann, Florian Pecune, and Justine Cassell. 2019. Faster responses are better responses: Introducing incrementality into sociable virtual personal assistants. In *9th International Workshop on Spoken Dialogue System Technology*. Springer, 111–118.

[34] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*. 1835–1844.

[35] Pontus Wärnestål, Lars Degerstedt, and Arne Jönsson. 2007. Interview and delivery: Dialogue strategies for conversational recommender systems. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA 2007)*. 199–205.

[36] Longqi Yang, Cheng-Kang Hsieh, Hongjian Yang, John P Pollak, Nicola Dell, Serge Belongie, Curtis Cole, and Deborah Estrin. 2017. Yum-me: a personalized nutrient-based meal recommender system. *ACM Transactions on Information Systems (TOIS)* 36, 1 (2017), 7.

[37] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 177–186.

[38] Ran Zhao, Alexandros Papangelis, and Justine Cassell. 2014. Towards a dyadic computational model of rapport management for human-virtual agent interaction. In *International Conference on Intelligent Virtual Agents*. Springer, 514–527.

[39] Ran Zhao, Oscar J Romero, and Alex Rudnicky. 2018. SOGO: a social intelligent negotiation dialogue system. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*. ACM, 239–246.