Conversational agent research toolkit: An alternative for creating and managing chatbots for experimental research

Araujo, T.

DOI
10.5117/CCR2020.1.002.ARAU

Publication date
2020

Document Version
Final published version

Published in
Computational Communication Research

Conversational Agent Research Toolkit

An alternative for creating and managing chatbots for experimental research[1]

Theo Araujo

CCR 2 (1): 35–51
DOI: 10.5117/CCR2020.1.002.ARAU

Abstract

Conversational agents in the form of chatbots available in messaging platforms are gaining increasing relevance in our communication environment. Based on natural language processing and generation techniques, they are built to automatically interact with users in several contexts. We present here a tool, the Conversational Agent Research Toolkit (CART), aimed at enabling researchers to create conversational agents for experimental studies. CART integrates existing APIs frequently used in practice and provides functionality that allows researchers to create and manage multiple versions of a chatbot to be used as stimuli in experimental studies. This paper provides an overview of the tool and provides a step-by-step tutorial of to design an experiment with a chatbot.

Keywords: conversational agents, natural language processing, content analysis, dialogue management, experimental designs

Conversational agents in the form of chatbots available in messaging platforms (e.g., Facebook Messenger, Skype, Telegram) are gaining increasing relevance in our communication environment (Araujo, 2018; Brandtzaeg & Følstad, 2017; Dale, 2016; Olmstead, 2017). Enabled by advances in natural language understanding and generation, a new set of conversational agents trained on increasingly larger and more varied datasets, makes use of computational methods to automatically interact with users in a wide variety of contexts.

These agents point to a major paradigm change in communication research (Zhao, 2006), shifting from the idea that technologies enable only computer-mediated communication among humans, to "the notion of the medium as the communicator" (Peter, 2017, p. 203). Their increasing adoption opens up a series of important research questions for communication research (Gunkel, 2012; Guzman & Lewis, 2019) regarding the content and consequences of interactions with these agents in contexts encompassing nearly all areas of communication, including health (Bickmore & Gruber, 2010; Bickmore & Picard, 2005; Ho, Hancock, & Miner, 2018), marketing (Araujo, 2018; Verhagen, van Nes, Feldberg, & van Dolen, 2014), news (Barot, 2017; Lichterman, 2016), and even politics (Woolley & Howard, 2016).

We present here a tool, the Conversational Agent Research Toolkit (CART), designed to help researchers create conversational agents (chatbots) for experimental studies aimed at exploring the content and consequences of human-machine interactions. To do so, CART provides a unifying toolkit written in Python that integrates existing services and APIs for creating and publishing chatbots as either a web interface or within messaging apps.[2]

CART acts as an integration layer across these different services, extending them with features aimed at simplifying the deployment and management of experimental studies. More specifically, CART allows the researcher to (1) create many (parametric) versions of a chatbot to be used in different experimental conditions, and have the chatbot interact differently with each participant depending on the condition, (2) randomly assign participants to these experimental conditions, (3) log all the interactions that participants have with the chatbot, storing these conversations along with participant-related information in a database readily available to the researcher, (4) integrate with survey flows, enabling participants to answer questionnaires before and/or after the interaction with the chatbot, and (5) integrate custom classifiers and override the APIs for dialogue management, providing an additional layer of control to the researcher, as outlined below.

Related Approaches

Three main approaches can be highlighted for investigating the content and consequences of interactions with conversational agents in an experimental setting: (1) use existing chatbots, (2) use the Wizard-of-Oz (WoZ)

method, or (3) creating one or more chatbots to be used as stimuli for the experiment using APIs or a toolkit such as CART. We briefly contrast the three approaches below.

Using existing chatbots

The first option available for researchers is to simply use existing chatbots to investigate the content and consequences of human-machine interactions, as done by earlier studies which have used for example the chatbot Cleverbot (e.g., Corti & Gillespie, 2016) or instructed the participants to visit websites of companies that have conversational agents and interact with them (e.g., Etemad-Sajadi, 2016). This approach presents lower levels of complexity to the researcher and potentially offers advantages when it comes to ecological validity by using existing agents, especially when the researcher is interested in investigating the effects of interactions with existing chatbots. However, it presents, we argue, four main limitations.

First, by using existing conversational agents, researchers are obviously limited to using one of the existing chatbots available elsewhere, having therefore no control over how the chatbot will interact with the participant, and potentially leading to shallow conversations – especially in cases where additional information (e.g., having actually an order in the case of customer service, or a reader profile in the case of chatbots for news) might be needed.

Second, researchers are restricted in terms of what the manipulations across experimental conditions will be. More specifically, because the researcher has no control over the existing chatbots being used for the experiment, the only manipulations across conditions that can be done are either (a) having participants complete different tasks with the same chatbot (by changing the instructions given to a participant) or (b) using different chatbots in each condition. This may pose challenges to the internal validity of the experiment.

Third, the logging of the conversations between the participant and the chatbot can be cumbersome and require workarounds, such as recording the session via screen captures and/or asking the participant to copy and paste the conversation back in a survey.

Finally, issues of privacy may arise, as the participant is interacting with a conversational agent from another platform and/or organization – and as such the participant data is being stored and processed by a third-party outside of the control of the researcher, without any clarity on terms of service or user agreement.

Wizard-of-Oz Method

A second option is to use the Wizard-of-Oz (Woz) method (Dahlbäck, Jönsson, & Ahrenberg, 1993), in which, generally speaking, participants in the experiment are told that they are interacting with a conversational agent, while they are actually chatting with human research assistants posing as automated agent. This method has been used in earlier studies (e.g., (Ho et al., 2018) and is often an alternative to either overcome the limitations of current technology, or because the infrastructure and knowledge required to deploy an actual conversational agent enabled by computational methods is not readily accessible to communication researchers. Whereas WoZ as a method might useful for several reasons and contexts (for an overview, see Dahlbäck, Jönsson, & Ahrenberg, 1993), we argue that exclusively relying on it may pose limitations to communication research for several reasons.

First, it is resource intensive, as research assistants need to interact individually with every participant, increasing the cost and the time required to do research, thus setting a limit to the number of participants that a study may be able to realistically include in an experiment. Second, extensive measures are needed to ensure that each conversation follows exactly the same script and flow, with the usage of research assistants therefore always presenting a potential risk that participants will be exposed to variations not intended by the research design or figure out that they are actually interacting with a human. Third, and perhaps most importantly in the long term, by not using the actual technology and instead simulating the whole interaction, researchers might not expose themselves to the computational methods available, thus restricting their ability to critically reflect upon the limitations and opportunities of this new technology.

Creating conversational agents for research

Finally, a third approach is to create conversational agents specifically designed to address specific research questions (e.g., Araujo, 2018; Zarouali, Van den Broeck, Walrave, & Poels, 2018). CART is positioned within this approach. Creating conversational agents can be done using one of the several technical solutions currently available, including a series of web services and API's (e.g., Facebook's Wit.ai, Google's DialogFlow, Amazon's Lex, Microsoft Azure Bot Service). Open source frameworks such as RASA for Python (Bocklisch, Faulker, Pawlowski, & Nichol, 2017) also exist, and can be installed directly on servers.

While these alternatives are relatively easy to configure, they generally lack direct functionality to setup experimental designs, store and report

conversations per participant, integrate the agent within an online questionnaire flow, and integrate custom classifiers. CART extends the functionality currently made available via APIs by acting as an integration layer, allowing the researcher to have control over the agent within an experimental setup. In its first release[3], CART uses DialogFlow as the dialogue manager, and Microsoft Bot Framework as the platform to publish the agent as a web chat, that can be embedded in a survey flow, or within messaging applications such as Skype or Facebook Messenger.

The toolkit

Intended purpose

CART allows researchers create text-based conversational agents for experimental studies, and to generate output that can also be analyzed using computational methods. As such, CART was designed to provide the following functionality:

Creation of multiple experimental conditions. CART offers the ability to create and manage multiple experimental conditions, in which the same conversational agent can randomly assign participants to different conditions. The agent then interacts differently with participants depending on the experimental condition, enabling researchers to compare the effect of different manipulations regarding what the conversational agent says to the participant, how the agent says it, or the overall flow of the dialogue. While researchers might be able to achieve the same objectives by creating different chatbots using an API, CART makes the process more streamlined in two ways. First, it simplifies this workflow by allowing the researcher to create only one agent using the API (e.g., DialogFlow) and handle differences across experimental conditions through CART, instead of having to create multiple agents (e.g., four agents in the case of a 2 x 2 design). Second, by making a distinction between the main dialogue in the API – that should be equal across conditions – and the manipulations that differ across conditions – that are managed via CART, this approach arguably provides more control and oversight to the researcher in terms of what is being manipulated, and makes the maintenance or changes to the agents potentially easier.

Dialogue management. Most of the dialogue between the conversational agent and the participant can be configured using an existing dialogue management API (DialogFlow) via its web interface, without the need of programming knowledge. This reduces the complexity of setting up the study, as most of the configuration can be done online.

CART extends the off-the-shelf functionalities from this API and pre-processes all the inputs and outputs of the dialogue according to custom conditions set by the researcher. This can be used, for example, to change specific utterances by the agent depending on the experimental condition in which the participant is in, or to override the dialogue management when the text typed by the participant meets specific criteria or patterns.

Integration with custom classifiers. CART allows the researcher to integrate custom functions, including (pre-trained) classifiers to the conversation flow. These custom functions can be used to modify the behavior of the agent depending on the user's input. For example, sentiment analysis can be applied to any response provided by the participant, and the results of the sentiment analysis may be used to redirect or control the response provided by the conversational agent – for example by providing a contextual response to a question such as "how are you today?". The results can also simply be stored along with the logs for future analysis.

Storage of conversation logs. CART ensures that all conversations between participants and the agent are stored in a database under the control of the researcher, thus enabling the execution of subsequent content analyses (manual, or automated) of the interactions between the participant and the agent, and the reporting of basic metrics out of the conversation (e.g., number of turns taken), or of the custom classifiers integrated to the flow. Existing dialogue management APIs do offer the option to store the history of conversations[4] yet exporting and consolidating this information per participant often requires workarounds, and, in some cases, researchers might actually want to turn off the logging to Google's or Microsoft's platforms to ensure privacy of the participants.

Integration with Online Surveys. CART enables the integration with common online survey platforms (e.g., Qualtrics) to allow for experiments that combine interaction with the conversational agent and self-reported measures. It can be setup in a way that the interaction between the participant and the agent can take place before, during, or after completing the online questionnaire. The integration ensures that a unique identifier is passed along between CART and the online questionnaire, allowing the researcher to link the questionnaire responses (self-reports) to the conversation logs between the participant and the agent.

Installing and using CART

CART is available as an open-source tool[5] with extensive documentation about installation and usage[6]. To use CART, researchers should have a basic understanding of Python, access to a server and a MySQL database, and accounts with the APIs used by CART[7].

Demo: Exploring the Influence of Anthropomorphic Features

To demonstrate the usage of CART, we show here how to create an agent for an experiment testing the extent to which anthropomorphic features of a conversational agent (e.g., a human-like name, or informal language style) may influence perceptions about the agent, as well as recommendation adherence. This agent is partially adapted from Araujo (2018), and theoretically investigates the influence of the social presence heuristic (Sundar, 2008) and anthropomorphism (Epley, Waytz, & Cacioppo, 2007).

The agent in this example automatically assigns participants to a condition, presents itself in different ways (with or without anthropomorphic features), asks some questions to the participant, and makes a recommendation. After the recommendation is made, the agent asks the participant whether she intends to follow the recommendation, and the reason why. Table 1 shows a proposed conversation path for the human-like and the machine-like condition.

The text that changes depending on the experimental condition is shown above in italics. As a note, the dialogue is simplified and being used as a demonstration only; an actual experiment would likely have a larger conversation flow, more complex conditions, and make sure that the agent actions are contingent on the participant responses – leveraging off-the-shelf capabilities from the dialogue management tool, or custom conditions or classifiers integrated via CART's flow.

All responses by the participant are stored in the conversation logs, and sentiment analysis is applied to each response. The agent is embedded in a larger questionnaire flow, so it starts asking the participant for a participant ID (generated by the online questionnaire tool) and provides a conversation code – which the participant needs to provide when continuing with the survey. The configuration files and the agent setup at the dialogue management tool are available online[8].

Step 1. Installing CART. The first step, covered in detail in the documentation[9], is to install CART in the server and connect it to the dialogue management tool and the MySQL database. After this step is completed, a basic agent is ready for customization.

Table 1. Conversation path for both conditions

Turn	Human-like	Machine-like
Participant	Hi	Hi
Chatbot	Hello! My name is *Ben*, I'm a virtual assistant to help you make nutritional choices for your breakfast. Before we start: what is your participant id?	Hello! My name is *NutriBot*. I'm a virtual assistant to help you make nutritional choices for your breakfast. Before we start: what is your participant id?
Participant	It's A123.	It's A123.
Chatbot	Great! Let's get started then. I need to know a bit more about you before I can make a suggestion. Are you male or female?	OK. The system needs some information about you before it can make a recommendation. Are you male or female?
Participant	Female	Female
Chatbot	*Gotcha!* And how old are you?	*OK*. And how old are you?
Participant	23.	23.
Chatbot	*Cool! And, just between the two of us,* how do you feel about your current weight?	*OK*, and how do you feel about your current weight?
Participant	I'm not very happy. I'd like to lose some weight.	I'm not very happy. I'd like to lose some weight.
Chatbot	*Thanks! Hey... so here's an idea for your breakfast. . .* quinoa fruit salad. It tastes great and has great nutritional value. Would you be willing to give it a try?	*OK. Based on your answers, the recommended breakfast is* quinoa fruit salad. It tastes great and has great nutritional value. Would you be willing to give it a try?
Participant	I guess so.	I guess so.
Chatbot	OK... could you tell me a bit why?	OK. . . could you tell me a bit why?
Participant	It seems like a good idea.	It seems like a good idea.
Chatbot	*OK! Thanks a million for chatting with me!* You can now continue with the survey. The conversation code that you need is B1505. *Have a great day!*	*Thank you.* You can now continue with the survey. The conversation code that you need is B1502. *Conversation ended.*

Step 2. Configuring the basic dialogue. The basic dialogue between the agent will be handled in a series of questions and answers, along one conversation path which, in DialogFlow. All the configurations are done via DialogFlow's web interface, without the need for programming knowledge. The dialogue will be managed with one single intent, named "Experiment". This intent is triggered when CART validates the participant id (see step 5) and provides the token [START_EXPERIMENT] to DialogFlow and ends when all the questions by the agent have been answered. Each question is configured as a separate parameter with the question by the agent included in the prompt, as outlined below. The sections of the response that will be different across conditions are substituted by a token (all upper case and between square brackets). Table 2 shows a potential configuration.

Table 2. Basic dialogue configuration

Parameter Name	Entity	Value	Prompt
Gender	@sys.any	$gender	[ACKNOWLEDGEMENT1] Are you male or female?
Age	@sys.any	$age	[ACKNOWLEDGEMENT2] And how old are you?
feelings	@sys.any	$feelings	[ACKNOWLEDGEMENT3] how do you feel about your current weight?
recommendation	@sys.any	$recommendation	[RECOMMENDATION] quinoa fruit salad. It tastes great and has great nutritional value. Would you be willing to give it a try?
Reason	@sys.any	$reason	OK... could you tell me a bit why?

The text response in DialogFlow closes the conversation. It can be configured as: [*CLOSURESTART*] *You can now continue with the survey. The conversation code that you need is* |*CONVERSATIONCODE*|. [*CLOSUREEND*].

The tokens (between square brackets) are also substituted by text depending the experimental condition, and the section |CONVERSATIONCODE| is substituted by a unique code that the participant can use when continuing to the survey, as outlined in step 5.

Step 3. Configuring the experimental conditions. After the basic dialogue is configured in the dialogue manager, CART can be used to assign participants to conditions, and replace the tokens by the actual manipulations. To do so, the config.yaml file needs to be edited in two sections. First, the section experimental_design needs to define that CART will assign participants to conditions using the random_balanced option – which randomly assigns participants to conditions while ensuring a balanced number of participants per condition – and specifies the conditions. For the example, it would work as shown in the code sample 1.

Code Sample 1. ***Configuration of the experimental design***

```
experimental_design:
  assignment_manager: CART
  assignment_method: random_balanced
  conditions:
    condition_1:
      condition_name: machine
    condition_2:
      condition_name: humanlike
```

Second, the rephrases section specifies how the agent should substitute the tokens configured in the dialogue management by the actual manipulations as shown in code sample 2.

Code Sample 2. ***Configuration of the experimental design***

```
rephrases:
  condition_1:
    AGENTNAME: NutriBot
    ACKNOWLEDGEMENT1: OK. The system needs some information
    about you before it can make a recommendation.
    ACKNOWLEDGEMENT2: OK.
    ACKNOWLEDGEMENT3: OK, and
    RECOMMENDATION: OK. Based on your answers, the recommended
    breakfast is
    CLOSURESTART: Thank you.
    CLOSUREEND: Conversation ended.
  condition_2:
    AGENTNAME: Ben
    ACKNOWLEDGEMENT1: Great! Let's get started then. I need to know
    a bit more about you before I can make a suggestion.
    ACKNOWLEDGEMENT2: Gotcha!
    ACKNOWLEDGEMENT3: Cool! And, just between the two of us
    RECOMMENDATION: Thanks! Hey... so here's an idea for your
    breakfast…
    CLOSURESTART: OK! Thanks a million for chatting with me!
    CLOSUREEND: Have a great day!
```

Step 4. Integrating a custom classifier: Sentiment analysis. In this example, every utterance by the participant is sent to Vader, a sentiment analysis classifier (Gilbert & Hutto, 2014), and the output of the classification is stored in the conversation logs. A function that receives the user message, executes the classifier, and returns a sentiment score is added to the special_functions.py file, and the config.yaml file is edited to integrate the function to the conversation flow as outlined in the code sample 3. The code specifies the name of the function (check_sentiment), the table in the database to store its output (logs), and the name (sentiment) and type (float) of the field that should be created in the database for this output. More advanced flows can also indicate criteria upon which CART will override the normal flow of the conversation based on the output of the classifier (with function_action being set to True)[10].

Code Sample 3. Integration of sentiment analysis

```
special_functions:
    function_1:
        function_name: check_sentiment
        store_output: logs
        store_output_field: sentiment
        store_output_field_type: float
        function_action: False
```

Step 5. Integrating with the questionnaire flow. The final step in the set-up is to integrate the agent within a questionnaire flow. In this example, the interaction between the participant and the agent will take place during the questionnaire, i.e., the participant will first go through some questions in the online questionnaire, and then be exposed to a page in which the agent is embedded as a web chat. In this configuration, the agent first asks a participant id – automatically generated code by the survey tool and displayed in the instructions provided to the participant –, interacts with the user, and provides a conversation code at the end of the conversation.

This integration is managed via three sections of the config.yaml. First, the initial section (other), configures the conversation code given by the agent at the end of the conversation. In this example, the conversation code always starts with a B (suffix), and the numbering starts at 1500 (to prevent that participants receive a very low number, such as B0). Second, the questionnaire_flow section indicates that the integration happens during the survey flow and provides a range of acceptable suffixes for participant id's (in this case, only participant id's starting with the letter A). The configuration also specifies what token CART should pass through to the dialogue management tool if the participant id provided is invalid[11]. Third, the connect_intents section indicates that when CART determines that the participant id is valid, the token START_EXPERIMENT should be used, to kick-off the experiment as configured in step 2.

An example of the configuration can be seen in the code sample 4, and figure 1 shows how this integration would be experienced by a participant.

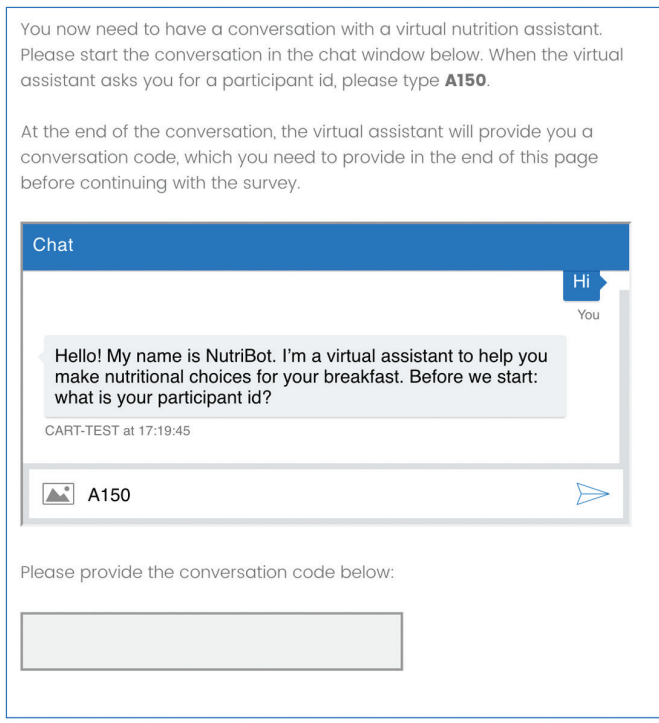
Code Sample 4. ***Integration of the survey flow***

```
other:
    (...)
    conversationcode_suffix: B
    conversationcode_base: 1500
```

```
questionnaire_flow:
    enabled: True
    moment: during
    config_during:
        rephrase_start_token: VALIDATEPARTICIPANTID
        participantid_dialog_field: participantid
        participantid_not_recognized: PARTICIPANTID_INVALID
        participantid_valid_suffixes: A

connect_intents:
    PARTICIPANTID_VALID: START_EXPERIMENT
```

Finally, the dialogue management tool needs to be configured with three additional intents. The first one, Welcome, is configured to start when the participant says "hi" (or related greetings) to the agent, asks for a participant id as a required parameter (Table 3), and has a text response the token [PARTICIPANTID_VALID]. When this intent is triggered, CART checks if it can find a valid participant id in the utterance by the participant and, if so,



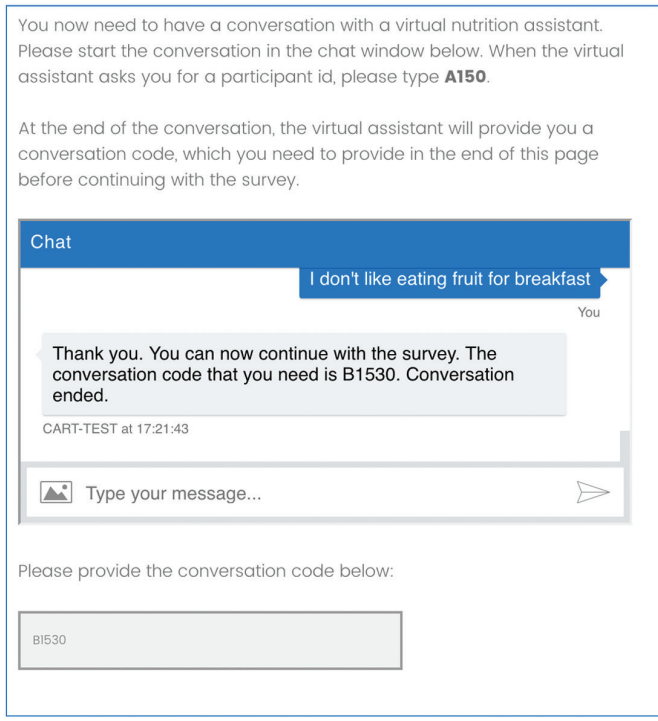


Figure 1. Survey and conversational agent integration

it allows the conversation to go to the experiment. If a valid participant is not found, the intent invalid participant id in DialogFlow is triggered with the token [PARTICIPANTID_INVALID] as the only training phrase, and its text response is the error message that the participant should receive when no valid participant id is found. Finally, a fallback intent (e.g., validate participant id) can also be configured to handle cases of participants that may want to try to provide a participant id again[12].

Table 3. Configuration of the validate participant id parameters

Parameter Name	Entity	Value	Prompt
participantid	@sys.any	$participantid	Hello! My name is [AGENTNAME]. I'm a virtual assistant to help you make nutritional choices for your breakfast. Before we start: what is your participant id?

Discussion

CART presents, in summary, a toolkit enabling communication science researchers to leverage computational methods to design conversational agents for experimental studies, and to generate output that can also be analyzed using computational methods. By managing experimental conditions, storing conversation logs, integrating with custom classifiers and integrating with survey flows, CART integrates to and extends existing dialogue management APIs, giving the communication researcher control over the conversational agent, and presenting an alternative to the WoZ method or to using existing chatbots as stimuli for experiments.

Limitations and Future Work

Some limitations, however, need to be acknowledged. First, CART is currently best suited for one-shot studies. While its underlying code has already been tested for longitudinal designs, in which participants need to interact with the agent several times, its first version would require extension by the interested researcher to do so. Future releases will include this functionality.

Second, CART uses the DialogFlow API for dialogue management, and the Microsoft Bot Framework to make the agent available in a web chat, or across messaging services. While this configuration was selected to minimize the learning curve and simplify its operation, researchers should be aware that CART is therefore limited to the core functionality of these systems, and subject to their terms and conditions. Future releases will also integrate with other open source frameworks (e.g., RASA) and API's.

Third, CART is primarily aimed at text-based conversational agents (e.g., chatbots). While the Microsoft Bot Framework allows for the integration of chatbots in voice-based virtual assistants (e.g., via Skype calls) making it possible for agents built in CART to be published in voice-based platforms, some features would need to be adapted for this modality of interaction (e.g., questionnaire integration).

Fourth, researchers interested in using CART should be aware that automated chatbots may be prone to errors in ways that using the WoZ method is not, and as such introduce different types of biases. Carefully piloting the chatbot, on the one hand, and inspecting the conversation logs to control for these biases, on the other hand, are therefore extremely important in these designs.

Fifth, researchers must consider the impact to their experimental designs of having the chatbot shift the way it responds based on user's earlier

responses (as, for example, with sentiment analysis being used to guide how the chatbot should respond). While the importance of contingency in responses has been demonstrated to be an important factor for downstream perceptions of conversational agents (e.g., Sundar, Bellur, Oh, Jia, & Kim, 2016), researchers should at least control for these shifts in the subsequent analyses – thus making the review of conversation logs even more important.

Finally, the researcher should be aware that informed consent needs to be requested by the participant – be it via the survey, or via the dialogue with CART – to ensure that the participant is fully aware that conversations are being logged and will be reused in the future. Researchers within the European Union should also carefully consider and discuss with Ethical Review Boards where the conversation logs are being stored, or being processed, in line with the General Data Protection Regulation (GDPR), and take steps to minimize the amount of (unnecessary) personal data being used in the study.

Notes

1 This research was supported by the Research Priority Area Communication and its Digital Communication Methods Lab (digicomlab.eu) at the University of Amsterdam. The author would like to thank the two anonymous reviewers for their constructive feedback on earlier versions of the paper.

2 In its initial version, CART uses the API of DialogFlow for dialogue management, and Microsoft Bot Framework for publishing the agent in a webchat and/or other messaging platforms (e.g., Skype or Facebook Messenger).

3 DialogFlow and Microsoft Bot Framework were selected in the first release due to the more intuitive web interfaces for configuring the basic dialogue management and for publishing the agent, and relatively lower requirements for server performance. At the time of writing, the standard edition of DialogFlow is offered free of charge by Google, and Microsoft Azure offers unlimited messages for free in standard channels, and up to 10,000 messages per month for free on premium channels. Researchers are advised to review the terms & conditions of these services before conducting an experiment.

4 E.g., https://dialogflow.com/docs/agents/history

5 https://github.com/uvacw/CART

6 https://cart.readthedocs.io/en/latest/

7 The documentation provides additional details about the requirements.

8 https://github.com/uvacw/CART

9 See section Installation and Setup Guide at https://cart.readthedocs.io/en/latest/

10 Not included in this example for conciseness, but available in the documentation at https://cart.readthedocs.io/en/latest/

11 The tutorial code available at https://github.com/uvacw/CART provides a configuration file for the DialogFlow agent used in this tutorial.

12 Full configuration not provided in this paper for conciseness. Agent configuration available in the tutorial folder at https://github.com/uvacw/CART

References

Araujo, T. (2018). Living up to the chatbot hype: The influence of anthropomorphic design cues and communicative agency framing on conversational agent and company perceptions. *Computers in Human Behavior*, 85, 183–189. https://doi.org/10.1016/j.chb.2018.03.051

Barot, T. (2017). The future of news is humans talking to machines. Retrieved March 9, 2018, from Nieman Lab website: http://www.niemanlab.org/2017/09/the-future-of-news-is-humans-talking-to-machines/

Bickmore, T., & Gruber, A. (2010). Relational Agents in Clinical Psychiatry. *Harvard Review of Psychiatry*, 18(2), 119–130. https://doi.org/10.3109/10673221003707538

Bickmore, T., & Picard, R. W. (2005). Establishing and Maintaining Long-term Human-computer Relationships. *ACM Transactions on Computer-Human Interaction* (*TOCHI*), 12(2), 293–327. https://doi.org/10.1145/1067860.1067867

Bocklisch, T., Faulker, J., Pawlowski, N., & Nichol, A. (2017). Rasa: Open source language understanding and dialogue management. ArXiv Preprint ArXiv:1712.05181.

Brandtzaeg, P., & Følstad, A. (2017, November 23). Why people use chatbots. Presented at the *4th International Conference on Internet Science*, Thessaloniki, Greece.

Corti, K., & Gillespie, A. (2016). Co-constructing intersubjectivity with artificial conversational agents: People are more likely to initiate repairs of misunderstandings with agents represented as human. *Computers in Human Behavior*, 58, 431–442. https://doi.org/10.1016/j.chb.2015.12.039

Dahlbäck, N., Jönsson, A., & Ahrenberg, L. (1993). Wizard of Oz studies: Why and how. *Proceedings of the 1st International Conference on Intelligent User Interfaces*, 193–200. Retrieved from http://dl.acm.org/citation.cfm?id=169968

Dale, R. (2016). The return of the chatbots. *Natural Language Engineering*, 22(5), 811–817. https://doi.org/10.1017/S1351324916000243

Epley, N., Waytz, A., & Cacioppo, J. T. (2007). On seeing human: A three-factor theory of anthropomorphism. *Psychological Review*, 114(4), 864–886. https://doi.org/10.1037/0033-295X.114.4.864

Etemad-Sajadi, R. (2016). The impact of online real-time interactivity on patronage intention: The use of avatars. *Computers in Human Behavior*, 61, 227–232. https://doi.org/10.1016/j.chb.2016.03.045

Gilbert, C., & Hutto, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media* (*ICWSM-14*). Available at (20/04/16) Http://Comp. Social. Gatech. Edu/Papers/Icwsm14. Vader. Hutto. Pdf.

Gunkel, D. (2012). Communication and Artificial Intelligence: Opportunities and Challenges for the 21st Century. *Communication +1*, 1(1), 1–25. https://doi.org/10.7275/R5QJ7F7R

Guzman, A. L., & Lewis, S. C. (2019). Artificial intelligence and communication: A Human–Machine Communication research agenda. *New Media & Society*, 1461444819858691. https://doi.org/10.1177/1461444819858691

Ho, A., Hancock, J., & Miner, A. S. (2018). Psychological, Relational, and Emotional Effects of Self-Disclosure After Conversations With a Chatbot. *Journal of Communication*. https://doi.org/10.1093/joc/jqy026

Lichterman, J. (2016). Alexa, give me the news: How outlets are tailoring their coverage for Amazon's new platform. Retrieved March 9, 2018, from Nieman Lab website: http://www.niemanlab.org/2016/08/alexa-give-me-the-news-how-outlets-are-tailoring-their-coverage-for-amazons-new-platform/

Olmstead, K. (2017, December 12). Nearly half of Americans use digital voice assistants, mostly on their smartphones. Retrieved December 27, 2017, from Pew Research Center website:

http://www.pewresearch.org/fact-tank/2017/12/12/nearly-half-of-americans-use-digital-voice-assistants-mostly-on-their-smartphones/

Peter, J. (2018). New communication technologies and young people: The case of social robots. *Youth and Media*, 203–217. Nomos Verlagsgesellschaft mbH & Co. KG.

Sundar, S. S. (2008). The MAIN model: A heuristic approach to understanding technology effects on credibility. *Digital Media, Youth, and Credibility*.

Sundar, S. S., Bellur, S., Oh, J., Jia, H., & Kim, H.-S. (2016). Theoretical Importance of Contingency in Human-Computer Interaction Effects of Message Interactivity on User Engagement. *Communication Research*, 43(5), 595–625. https://doi.org/10.1177/0093650214534962

Verhagen, T., van Nes, J., Feldberg, F., & van Dolen, W. (2014). Virtual Customer Service Agents: Using Social Presence and Personalization to Shape Online Service Encounters. *Journal of Computer-Mediated Communication*, 19(3), 529–545. https://doi.org/10.1111/jcc4.12066

Woolley, S. C., & Howard, P. N. (2016). Political Communication, Computational Propaganda, and Autonomous Agents — Introduction. *International Journal of Communication*, 10(0), 9.

Zarouali, B., Van den Broeck, E., Walrave, M., & Poels, K. (2018). Predicting Consumer Responses to a Chatbot on Facebook. *Cyberpsychology, Behavior, and Social Networking*, 21(8), 491–497. https://doi.org/10.1089/cyber.2017.0518

Zhao, S. (2006). Humanoid social robots as a medium of communication. *New Media & Society*, 8(3), 401–419. https://doi.org/10.1177/1461444806061951

About the author

Theo Araujo works at the Amsterdam School of Communication Research (ASCoR)
University of Amsterdam.
Correspondence address: t.b.araujo@uva.nl