Empowering K-12 Students to Understand and Design Conversational Agents: Concepts, Recommendations and Development Platforms

by

Jessica Van Brummelen

B.A.Sc., The University of British Columbia (2017)
S.M., Massachusetts Institute of Technology (2019)

Submitted to the Department of Electrical Engineering
and Computer Science in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2022

Author . . . .
Department of Electrical Engineering and Computer Science
August 26, 2022

Certified by . . . .
Professor Harold Abelson
Class of 1922 Professor of Computer Science and Engineering
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Empowering K-12 Students to Understand and Design Conversational Agents: Concepts, Recommendations and Development Platforms

by

Jessica Van Brummelen

Submitted to the Department of Electrical Engineering and Computer Science
on August 26, 2022, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Conversation influences nearly every aspect of human life, and has done so for ages. Recently, people have begun to converse with technology. This method of technology interaction has rapidly become prominent, and raises unique questions about human-computer interaction. For instance, how do such human-like, relational interactions affect people's trust of computer systems? Researchers have started to investigate such questions with respect to adults, finding correlations between trust and anthropomorphism of agents. However, very little research investigates children's perceptions of these devices, and even less investigates how interventions might change these perceptions. This is despite how educational interventions have previously changed how people perceive and trust other technology. It is also despite how conversational technology is uniquely positioned to appeal to children, influence them relationally and potentially spread misinformation.

This dissertation presents educational interventions for K-12 students, which aim to encourage healthier understanding and relationships with conversational agents. This includes conversational agent curricula, development platforms and conceptual frameworks. Through studies with children and parents from Western, Industrialized, Educated, Rich and Democratic (WEIRD) and non-WEIRD countries, I found different subsets of the participants' perceptions of agents changed differently through the activities. For instance, after learning to program agents, participants from non-WEIRD countries felt agents were more competent, more dependable and more like authority figures than those from WEIRD countries did. Children consistently felt agents were warmer and more human-like than parents did. When participants discussed their trust of agents, I found they frequently mentioned where agents obtained their information, what agents do with the information they are given and how agents are programmed. I also found participants most often mentioned learning something when discussing why their trust changed.

These studies, as well as a systematic literature review and an analysis of various agent development platforms, informed the creation of a pedagogical framework of forty foundational conversational agent concepts, seventeen conversational agent design recommendations and thirteen conversational agent K-12 pedagogy recommendations. For instance, I recommend designing agents with more task-orientation in general, while considering the end user audience. I also recommend informing end users about the trustworthiness of agents through agent design and educational interventions. This is to increase transparency and allow end users to calibrate their trust accordingly. Educators may increase agent transparency through teaching the foundational agent concepts in the framework, which fall under the categories of natural language understanding; conversation representation; dialog management; data access and conversation context; and human-agent interaction. With conversational agents becoming increasingly ubiquitous, it is increasingly important for users of this technology—including and especially children—to be able to calibrate healthy perceptions and levels of trust towards it. This research aims to empower children to do this through a conversational agent design and pedagogy framework.

Thesis Supervisor: Professor Harold Abelson
Title: Class of 1922 Professor of Computer Science and Engineering

Acknowledgments

MIT is known by the world for its deep thinkers and creative makers; it is known by its students for its deeply supportive teachers, mentors and friends. It has been an incredible privilege to meet people here from so many different places, working on so many different projects, who all have one united goal: to develop technology to make a better world. I would like to thank my mentors, friends and family from within and without the MIT community, who have supported me in this shared goal.

First, to my committee: Hal Abelson, Cynthia Breazeal and Arvind Satyanarayan. Hal, thank you for supporting me in this five-year journey. I still remember first meeting you, and being amazed at your heart for empowering children to understand and develop meaningful technologies. This continues to be—and I believe, will always be—an inspiration to me. This is one of the greatest gifts a supervisor can give to their students. Thank you.

Cynthia, I feel incredibly fortunate to have learned from your deep knowledge of the K-12 AI educational space and relational technology—from the course where Zhorai was first born, to now having you on my committee. I also feel incredibly fortunate to be able to look up to you as a female researcher who is doing amazing work with technology and children. I cannot wait to see what you do next.

Arvind, I have learned so much from your deep knowledge of the human-computer interaction space, from your supportive and thoughtful feedback, and from the many thought-provoking campus events you have organized for HCI students. Thank you for supporting me—and so many other students—in improving how we interact with computers.

This research would not have been possible without the support of many staff members and collaborators; undergraduate and graduate student researchers; research scientists and professors; and K-12 students, parents and teachers. Through encouraging me as I transitioned from mechanical engineering to computer science; teaching me how to not only be a computer science student, but also a computer science researcher; programming late into the night with me; preparing studies with me; engaging in studies with me; and supporting me in so many other ways—you truly made this research possible. To name a few: Natalie Lao, Phoebe Lin, Daniella DiPaola, Randi Williams, Safinah Ali, Blakeley Payne, Nicole Pang, Xiaofei Zhou, Vik Tabunshchyk, Claire Tian, Nghi

Nguyen, Maura Kelleher, Liz Harkavy, Tommy Heng, Terryn Brunelle, CC Manigar, Yulia Gonik, Catherine Yeo, Kevin Weng, Helen Li, Mahmoud Khalifa, Funing Yang, Jessica Zhu, Daniel Kim, Jimin Lee, Evan Patton, Jeff Schiller, Susan Lane, Li Li, David Kim, Selim Tezel, Karen Lang, Robert Parks, Marisol Diaz, Cindy Rosenthal, Mark Friedman, Jeff Freilich, Lyn Turbak, Josh Sheldon, Mike Tissenbaum, Mark Sherman, Drew Nichols, Farzeen Harunani, Ilaria Liccardi, Irene Lee, Mel Robinson, Taylor Reynolds, Anni Rowland-Campbell, Ibrahim El Badawi, Leanne Fry, Kat Glenn and many others.

Finally, I would like to thank my friends and family, who have supported me through the many ups and downs of five years of study. You have provided me with housing (for months on end during a global pandemic!), patiently listened to me as I discussed my research, humoured me as I learned to ice skate and skateboard on campus, joined me for Canadian thanksgiving year after year, cared for me through medical situations, adventured with me across the continent, encouraged me through a degree I never thought I would even attempt, and brought me so much joy throughout the years. You all have made this period of my life not only manageable (because doing a PhD can be hard at times), but truly incredible.

A special thanks goes out to my partner, Morgan Paulgaard. You renewed my love for learning when I needed it most. Thank you for your endless curiosity, depth of care and wholehearted encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Our children are the rock on which our future will be built, our greatest asset as a nation. They will be the leaders of our country, the creators of our national wealth who care for and protect our people.
—Nelson Mandela

Conversational artificial intelligence (AI)—or the ability of a computer program to understand human language and respond accordingly—is ripe with potential. Imagine a conversational agent engaging children in learning history with a virtual Rosa Parks; an agent writing a persuasive article for you after you have a natural-language discussion; or an agent providing constant, accurate healthcare answers to those in need. With recent major advances in natural language processing and automatic speech recognition, including transformers, transfer learning and foundation models, these ideas are not far-fetched [198, 32, 46, 76, 90, 68, 28]. Furthermore, with sociotechnical imaginaries of conversational agents—influenced by agents in the media, like HAL, J.A.R.V.I.S., and the Star Trek computer—highly prevalent today, it is not difficult to envision natural language becoming the primary way we communicate with computers [80, 165, 158, 18].

Nonetheless, current agents, like Google Home, Apple's Siri and Amazon Alexa, do not live up to our imaginaries' ideals. Speech is misrecognized, with Alexa mistaking porches for Porsches, living rooms for Pudding Rooms, and Pampers for cancer [145, 183]; and natural language is misunderstood, with Siri suggesting nearby liquor stores after being asked what to do about alcohol poisoning [20]. These imperfections are not merely small inconveniences, but rather illustrate deeply-rooted flaws and biases in automatic speech recognition and natural language understanding that have serious, far-reaching consequences [149]. For instance, researchers found that widespread industry speech recognition systems by Amazon, Google, IBM and Microsoft did substantially worse when recognizing black speakers versus white [86]. Others have found significant gender biases in word embeddings, causing words like "homemaker" to be heavily associated with "woman", and "computer programmer" with "man" [213, 27]. Biases in AI systems are widespread, being found in medical technologies, justice systems, advertising, policing systems, and elsewhere [149].

Speech interfaces have also been critiqued for their façade of naturalness, as their designs are—in current reality—constrained and cumbersome compared to human-human interactions [21, 127, 85]. In 2010, Norman described how humans need to learn interface standards before any interactions can become truly natural. This includes learning speech interaction standards [127]. One challenge with nascent technologies like conversational interfaces, is that they often are not developed according to a set of such standards. Thus, end users have trouble developing accurate mental models of how the technology works, leading to frustration or dwindling satisfaction [21].

Nonetheless, because conversation is one of the most intuitive, primary methods humans use to communicate with each other, conversational interfaces are uniquely positioned to inspire relational interactions with technology [120, 160]. For instance, when the ELIZA agent was first developed in the 1960's, one of the developer's secretaries famously asked for privacy when talking to the agent—despite the algorithm's artificiality [156]. This agent has now won a Peabody Award for enabling software to engage in "emotional interactions, empathy, and connection" [39]. Furthermore, researchers have found that human-agent relationships can be modeled through Knapp's Staircase in the same way human-human relationships can be modeled. They also found correlations between this human-agent relationship development and increased trust of agents [160]. Considering how trust is a key factor in misinformation spread [210, 159], how agents could be used to circulate disinformation and reinforce human biases, and how associated risks could be particularly acute with children, conversational agent trust is becoming a serious concern for researchers and policy makers alike [182].

With other technologies, researchers have shown how educational interventions with users can affect trust, increase understanding and decrease the spread of misinformation [44, 200, 159, 166, 47]. For instance, DiPaola found children trusted robots less after engaging in societal impact curriculum [47]. Another study found children's trust decreased after learning about the programmatic nature of robots [166]. Despite the significant connections found between conversational interactions and trust, however, there is very little literature on educational interventions concerning conversational agents in particular—and even less involving children [191, 116, 115, 160]. That being said, there is literature on utilizing conversational agents in educational settings, teaching conversational agents new concepts, and investigating the various components of conversational agents [81, 3, 98, 142, 33, 7]. There is also literature on how to teach AI to a K-12 audience effectively [178, 100, 214] (as described in Section 1.2.1 and Chapter 5) and how to teach natural language processing, which is a key component of conversational AI [128, 146] (as described in Chapter 4).

Through engaging with such literature and K-12 pedagogy, as well as implementing studies with K-12 students, this dissertation develops an educational framework to teach K-12 students about conversational agents. This framework utilizes constructivist and constructionist learning theories to both empower students to understand technology and become technology developers [135, 70]. As we move towards a world filled with conversational potential, empowering people—especially from a young age—with deep understanding of agents' flaws and utility, as well as effective design principles will be vital to agents being used and developed wisely in the future. It can also empower young people to make positive change in their communities. In the workshops described in this dissertation, students developed agents to teach their friends a new language, help their grandparents remember words, and encourage their classmates to recycle, among other inspiring projects [186, 191, 193].

Ideally, agents should be developed to portray the reality of their abilities and limitations to their users through effective speech and conversation design. In a study with AI decision-aid agents, researchers describe the concepts of algorithmic aversion and algorithmic appreciation. They note that if users are too averse to agents' advice and information, they cannot truly benefit from using the agents. However, if they are too appreciative, they may make ill-informed decisions when agents present incorrect advice or information [59]. By portraying conversational agents in a realistic, truthful way through design, discrepancies between users' expectations of agents—or their agent "partner models"—and the reality of agents can be reduced, which can also reduce user frustration [49]. The K-12 framework in this dissertation aims to empower students to perceive and develop conversational agents in this way, by providing them with agent design recommendations, as well as a conceptual framework for understanding agents.

1.0.1 Developing the K-12 Conversational Agent Framework

To develop the design recommendations for this framework, I completed a survey of conversational agent usability design literature using a snowballing survey approach [208]. I describe this survey in Chapter 2. The survey identified current, prominent conversational agent design recommendation literature; however, the literature lacked recommendations for diverse users, such as those from countries that are not Western, Educated, Industrialized, Rich and Democratic (WEIRD) [167, 160, 117]. It also lacked recommendations regarding human-agent trust and perceptions.

A key goal of the K-12 framework is to empower nearly anyone—including those not well represented in the literature, like those from non-WEIRD countries—to foster healthy perceptions of agents and develop conversational agents to address their needs. To address this gap, I completed a study with K-12 students and their parents from WEIRD and non-WEIRD countries (described in Chapter 3). In this study, I investigated initial perceptions of agents, such as participants' levels of trust and partner models of agents, and how these perceptions changed through programming and societal impact activities. For example, after programming, children trusted agents significantly more than parents did, and participants from non-WEIRD countries thought agents were significantly more competent, authoritative and dependable than those from WEIRD countries did. Based on the results, I developed recommendations for designing agents as well as educational interventions. These recommendations included designing agents and interventions to foster appropriate levels of trust and partner models with respect to particular user-groups.

Through analyzing how participants' perceptions changed, as well as the educational activity itself, I developed additional recommendations for teaching conversational agent educational interventions to K-12 students. In Chapter 5, I note how teaching K-12 students requires unique approaches and presents unique challenges [214, 188]. Using the recommendations from my study results, as well as the results from a literature review of

K-12 AI education and a co-design study with K-12 teachers (which I completed alongside others from the AI education community) [214, 188], I developed a set of thirteen K-12 teaching guidelines. These teaching guidelines are for educators and researchers who would like to empower K-12 students to develop healthy understandings and perceptions of conversational agents. They include methods for how to increase engagement for K-12 students, scaffold their learning, develop conversational agent-specific competencies, and improve equity, diversity and inclusion.

For the final portion of the framework, I analyzed conversational agent development platforms and related educational materials to identify conversational agent concepts fundamental to understanding how they work. The concepts progress from those learned by interacting with agents (e.g., Alexa [193] or the Zhorai agent [99]) to those learned from developing simple agents (e.g., using ConvoBlocks [191] or Convo [215]) to those learned from creating robust, multi-turn agents (e.g., through Dialogflow CX [62]). The concepts fall into five categories, which were derived from conversational agent literature [142, 33, 7, 178, 100, 99, 185, 215, 14, 173]: (1) Natural Language Understanding, (2) Conversation Representation, (3) Dialog Management, (4) Data Access and Conversation Context and (5) Human-Agent Interaction.

By developing this conversational agent framework, this dissertation addresses gaps in conversational agent education [115, 116, 191] and democratizes this technology [185, 186, 191, 193], making it accessible to those without advanced computer sciences degrees. Specifically, I aim to empower K-12 students with the knowledge and skills necessary to foster healthy perceptions and trust of conversational technology, as well as empower them to create agents that do the same. All in all, this dissertation aims to answer the following main research questions.

RQ1.0.1: What are the most prominent design guidelines in current conversational agent literature, and how can they be taught in the K-12 context? (Chapters 2 and 5)

RQ1.0.2: What are the conversational agent concepts (e.g., intents, entities, training, etc.) taught by prominent conversational agent development tools and related educational materials (e.g., Alexa Skills Kit [10], Dialogflow [62], etc.), and how can they be taught in the K-12 context? (Chapters 4 and 5)

RQ1.0.3: How do people of various backgrounds (WEIRD and non-WEIRD, as well as different generations) perceive conversational agents while developing them, and what are the implications of this for K-12 conversational agent pedagogy and design recommendations? (Chapters 3 and 5)

RQ1.0.4: How do children and those without advanced computer science degrees envision the future of conversational agents, and what are corresponding forward-looking conversational agent design guidelines and concepts? (Chapter 3)

A brief overview of the results of investigating these questions follows. An in-depth summary can be found in Chapter 6.

- With respect to **RQ1.0.1**, I identified a prominent, comprehensive set of design guidelines, which were based on an exploration of existing graphical user interface and conversational agent guidelines. These guidelines are from Murad et al.'s paper, "Design Guidelines for Hands-Free Speech Interaction" [117].
- With respect to **RQ1.0.2**, I identified 40 fundamental conversational agent concepts taught by prominent conversational agent development tools and curriculum. These concepts fall into five categories: (1) Natural Language Understanding, (2) Conversation Representation, (3) Dialog Management, (4) Data Access and Conversation Context and (5) Human-Agent Interaction.
- With respect to **RQ1.0.3**, I found significant differences between how people from WEIRD and non-WEIRD countries, how children and parents, and how people from different intersectional groups perceive conversational agents with respect to trust and partner models. I also found significant differences before, during and after the programming and societal impact activities. For example, prior to the activities, females from WEIRD countries were more skeptical of agents than males were; however, there was no significant difference for females versus males from non-WEIRD countries. Based on these results, I developed design recommendations for conversational agents and educational interventions to encourage healthy perceptions and levels of trust of agents, with respect to particular user-groups.
- With respect to **RQ1.0.4**, I found participants overall envisioned their ideal conversational agents as generally being task-oriented, being balanced in terms of artificiality and human-likeness, and having abilities including useful, common features; user-oriented features; fun features; and emotionally intelligent features. Thus, I recommend developing agents with respect to these general characteristics and fea-

tures, and also considering the preferences of the particular end user audience. For instance, children and those from non-WEIRD countries described their ideal agents with relatively more social orientation and human-likeness than overall participants, and vice-versa for parents and those from WEIRD countries.

- With respect to **RQ1.0.1**, **RQ1.0.2** and **RQ1.0.3**, I developed recommendations for teaching conversational agent concepts and design to K-12 students. These include increasing student engagement, scaffolding student learning, empowering students to learn particularly challenging conversational agent-specific concepts, and focusing on equity, diversity and inclusion, among other recommendations.

1.1 Overview of Conversational Agent Tools and Studies

For this dissertation, I developed three K-12-focused educational software tools (as shown in Figure 1-1), which I used to investigate current perceptions of the technology, educate people on the capabilities and limitations of the technology, and empower teachers to incorporate AI education into their curricula. Through user studies with the tools, I developed design recommendations for creating effective educational conversational AI technology and related curricula. This informed the pedagogical conversational agent framework in this dissertation, and involved four main projects, as follows:

- **ConvoBlocks** (aka the MIT App Inventor Conversational Agent Platform): This block-based programming interface allows nearly anyone—including K-12 students—to program conversational agents. These agent programs run on Amazon Alexa devices and can communicate with mobile applications developed in MIT App Inventor [186, 185, 189, 191, 193].
- **Zhorai**: Zhorai is a teachable conversational agent. Through the process of teaching Zhorai about animals, students can learn about machine learning training, representation, classification and societal impact of AI [99].
- **Convo**: This agent allows people to program by speaking or typing in natural language. For example, someone might say, "If *foo* is greater than 5, say 'hooray' ", and Convo will convert the speech into Python code [215, 190, 184].
- **Teacher Co-design**: This project consists of activities for K-12 teachers to learn

about AI and develop AI curricula for their own subject (e.g., AI curricula for social studies, English, math, etc.). Teachers may use the conversational agent tools outlined above, or others, like Teachable Machine [34] or ML4Kids [91], within their designed curricula [188].

Figure 1-1: The three main conversational AI tools I developed for this dissertation: ConvoBlocks, which is a block-based coding tool to create conversational agents (left), Zhorai, which is a teachable conversational agent for K-12 machine learning education (middle), and Convo, which is a conversational agent users can talk to in natural language to program (right).

Both ConvoBlocks and Convo empower students to develop their own conversational agents. Compared to commercial conversational agent development interfaces, including Amazon's Alexa Developer Console [12] and Google's Dialogflow [62, 63], ConvoBlocks and Convo are targeted at a younger or less experienced audience and for a different purpose. Dialogflow and the Alexa Developer Console are platforms to enable software developers create commercially viable agents [62, 63, 12], whereas our systems were developed for K-12 students to create agents to solve problems in their personal communities while learning how to program [209, 186, 190]. I compare these systems and the conversational agent concepts they present to learners in Chapter 4, contributing to the K-12 pedagogical framework of foundational conversational agent concepts.

1.2 Background and Related Work

This section describes related work, including K-12 AI education initiatives, conversational agent design recommendations, tools to develop conversational agents, examples of conversational AI in K-12 education, and research involving perceptions of technology.

1.2.1 K-12 AI Education

Researchers are developing education initiatives to empower nearly anyone to understand AI and its societal implications. These initiatives are not limited to teaching adults, but rather, many focus on teaching children, as these children will soon be the policy makers, developers and consumers of this technology. For example, the AI4K12 initiative is developing tools and curricula based on five core "Big AI Ideas" specifically for the K-12 classroom [178]. MIT's Responsible AI for Social Empowerment and Education (RAISE) initiative is developing both vocational and K-12 tools and programs for AI education [111]. This growing community of researchers and educators is also not limited to North America [130, 69, 40, 201] or educational institutions [144, 141]. Researchers and entrepreneurs worldwide are engaging children in understanding the AI in our homes, workplaces, and even transportation systems [48, 191, 147].

There is also a related, growing body of literature emphasizing empowering students to develop technology to have authentic impact on their communities, or take "computational action" [176, 177, 95, 134]. Researchers posit that doing so provides increased learning motivation as well as a more inclusive learning environment [176, 177]. For example, young Mumbai students from *Girls for Change* created an Android app to help with water supply management [82]. Other students have developed machine learning models to investigate racial stereotyping in Tweets [95]. Often, people from underrepresented minority groups, like these students, have increased motivation when developing projects personally meaningful to them [177, 4]. K-12 AI education initiatives often take computational-action-related approaches to encourage diversity and equity, as well as engagement in their curricula [186, 215, 134, 100, 178].

In my own research, students have taken computational action through developing conversational agents to help others in their communities. They have created agents to teach people sign language, improve mental health, and help diagnose illnesses [191]. To facilitate learning—especially for underrepresented minorities, like females and those from non-WEIRD countries—I embedded computational action activities in my educational interventions in this dissertation (see Chapter 3). Through these interventions, students' confidence in terms of feeling like programmers, creating their own technology projects and being able to make an impact in their community significantly increased. Thus, in Chapter 5 I recommend including computational action activities in conversational agent educational curricula.

Many of the AI education initiatives have developed tools to facilitate K-12 AI learning and computational action [178, 100]. These tools range from educational, computer-vision enabled robots [179] to AI extensions for block-based coding environments [83, 112, 170, 1] to machine learning clustering visualization web tools [202]. Many of the tools also have associated curricula. For example, the "Personal Image Classifier", one of the block-based coding AI extensions, engages high school students in creating image classifiers and learning about machine learning [170]. Another AI teaching tool, "Popbots", facilitates learning the "Big AI Ideas" through engaging preschool students in interacting with and training social robots [206, 178].

These tools and associated curricula in the AI education space often utilize constructivist and constructionist learning theories, in which students create knowledge, often through developing projects or solving problems [70, 135, 191, 214]. Both learning theories can improve student engagement and are generally applicable to computer science education [97]. Computer science education has also benefited from a pedagogical framework of concepts called "computational thinking skills" [108, 31]. More recently, AI educators have realized there are competencies unique to AI that are important for students to learn. Researchers have built on the computational thinking skills framework [189], as well as developed their own frameworks to address this gap [100, 178]. In Appendix A, Table A.1 shows a list of AI competencies from a prominent framework by Long and Magerko [100]. These AI competencies, as well as AI education resources, which teach these competencies, provide structure for developing further AI teaching tools and resources, including conversational agent pedagogy. Alongside Zhou and Lin, I completed a literature review of such AI education resources, analyzing which AI competencies the resources addressed [214]. As shown in Appendix A, none of the resources addressed all competencies. Having this list of competencies, however, enables educators to identify resources to help them teach AI comprehensively.

In a similar way, there exists a need for a framework of conversational agent-specific competencies—especially considering conversational AI's unique positioning in terms of market penetration and potential to be a primary mode of human-computer interaction [165, 158, 116]. One of my studies also emphasized this need for additional conversational AI pedagogy, as there was evidence for students having more difficulty learning conversational AI-specific concepts than general AI concepts [191]. The AI literacy framework also emphasizes how it is important to understand a breadth of specific types of AI, such as conversational AI, through its competency, "Interdisciplinarity" [100]

Despite this need and agents' prevalence in young people's lives today, there is very little research in teaching conversational agent concepts—especially to young people [116]. One example includes a study by DiPaola, in which students learned about social robots. Specifically, they learned about the societal impact of social robots, and how to prototype robot conversation through a flow editor. The curriculum provides a broad overview of social robotics, including a portion on conversational AI, in which students learn concepts including conversational flow representation and machine learning [47]. It is intended for an audience aged 9-12. The curriculum from the study I present in Chapter 3 is intended for a slightly older audience of students aged 11-18. It teaches related concepts, as well as concepts like large language models, transfer learning and agent modularization, specifically focusing on conversational AI rather than social robotics. Other pedagogy utilizes conversational agents as a means for teaching other concepts, like the Zhorai activity [99] or Betty's Brain [26], rather than conversational agent concepts. Nonetheless, these activities can also unintentionally teach conversational agent concepts, as described in Section 1.2.3 and Chapter 4.

Although each of these examples teach conversational agent concepts, there is still a need for a comprehensive framework to provide students with a more nuanced and complete understanding of conversational agents; educators with additional structure when teaching about conversational agents; and researchers with a framework for assessing conversational agent understanding. In the next section, I describe this need with respect to design frameworks related to such conversational AI pedagogy. In later sections, I describe DiPaola's study in more detail with respect to investigating students' trust of agents and relationship modeling.

1.2.2 Design Frameworks and Tools for Teaching Conversational AI Development

Design guidelines and frameworks have been essential to developing effective user interfaces for decades [125, 122]. Although these types of guidelines have most often been developed for graphical user interfaces in the past, many are general enough to give some insight into effective conversational agent design. For instance, Hutchins et al. utilizes the idea of interface directness to identify gaps between user mental models and reality. According to Hutchins et al., an interface increases in directness, and thereby usability, when developers reduce semantic and articulatory distances [79]. For instance, developers might reduce semantic (or "meaning") distance by enabling agents to couple the meaning of end users' phrases directly to intended actions, like connecting the phrase "Turn on the kitchen lights" to the action of providing electricity to smart lights in a kitchen. Alternatively, there would be increased semantic distance if the same phrase was connected to the action of turning all the lights on in a house, since this was not the end user's intended meaning. Another general method for analyzing user interfaces includes utilizing Green's cognitive dimensions of notation. These dimensions can provide insight into the unique environment speech-based agents provide, and how they may not be appropriate for every use-case [66]. For example, purely speech-based agents may not be appropriate for design scenarios, since users cannot continuously observe structures being developed with speech, unlike with graphical user interfaces. To address such usability challenges, designers can also use standardized, agent-specific design heuristics.

Many different researchers have started to develop such agent-specific guidelines over the past few years [119, 118]. For instance, Murad et al. developed a set of voice user interface heuristics by surveying graphical user interface and voice user interface-based literature [118]. The same researchers then developed voice user interface heuristics specifically for graphical user interface-trained designers, such that they can easily shift to designing with the new interaction mode [119]. Other researchers have used very different foundations for their voice user interface design guidelines. For instance, Axtell and Munteanu developed voice user interface guidelines through a thorough analysis of the Star Trek agent, "Computer". The authors theorized that by developing agents similar to "Computer"—a likely source for the sociotechnical imaginaries of conversational agents today—users' expectations for such agents will be closer to reality [18]. Other researchers have utilized interviews and relevant theories, like self-determination theory or values-based design, to determine conversational agent design guidelines or interventions [212, 38].

Evidently, recent developments in conversational design have resulted in many variations on design guidelines. There exists a need to identify a comprehensive, prominent set of these guidelines, and integrate them into related pedagogy—especially since there is a current lack of voice user interface design in human-computer interaction courses [116, 115]. Through examining human-computer interaction courses at 25 of the top universities with respect to human-computer interaction publications, Murad and Munteanu found that only two of them had any curriculum specifically teaching voice user interface-related design [116]. In a literature review of child-agent human-computer interaction research, Garg et al. did not note any literature teaching children about conversational agents or how to design them, despite investigating educational conversational agent literature and agent design literature [58].

Other educational materials relevant to voice user interface design include guides from conversational agent development platforms, such as those created by companies like Google and Amazon. These materials typically involve tutorials, documentation and suggestions for conversational agent design. They are typically focused on teaching developers with prior programming knowledge and experience, rather than those with little prior knowledge [11, 64]. A number of my studies also include educational materials for agent design and development, which alternatively focus on those with little-to-no prior knowledge [191, 215, 186]. Other opportunities for learning about conversational agents include simply interacting with agents, like Alexa or Zhorai [99].

Each of these systems and educational materials present learning opportunities for people with differing levels of prior knowledge. This dissertation develops a framework of conversational agent concepts and usability design guidelines to empower people of differing prior knowledge to learn about and how to develop usable, effective agents. In Chapter 2, I complete a systematic literature review to identify a prominent, comprehensive set of usability guidelines for conversational agent development. In Chapter 3, I develop additional guidelines with respect to the results of my study with children and adults from WEIRD and non-WEIRD countries. In Chapter 4, I analyze conversational agent development platforms and educational materials to identify foundational conversational agent concepts. Finally, in Chapter 5, I develop guidelines to teach these concepts and usability guidelines effectively. Each of these chapters contribute to the K-12 pedagogical conversational agent framework, addressing the gap in conversational agent design education in the literature.

1.2.3 Conversational AI and K-12 Students

In this section, I describe literature related to conversational agents and young people in K-12 education. There is little literature related to teaching K-12 students conversational agent concepts; however, there is literature related to using agents as teaching tools and observing K-12 students perceptions of agents, as described in the following subsections.

Tools to Create Conversational Agents

There are few dedicated tools to teach people conversational AI, and even fewer to teach young people [185, 58]. One tool includes ConvoBlocks, which is described in this thesis. It was specifically developed to teach conversational AI concepts and empower young students to develop complex, industry-level conversational agents. Another related tool includes the Interaction Flow Editor, which DiPaola used to teach students about social robot conversation design. This editor abstracts away many of the low-level details, which allowed students ages 9-12 to develop conversations [78, 47]. Conversely, ConvoBlocks provides students with a less-abstracted, yet still accessible method to program agents, and aims to teach students in middle and high school. Specifically, its method of block-based coding teaches students the low-level programming concepts of functions, conditionals and events, while still abstracting away syntax errors through puzzle-block-like code [186].

Other related tools, including Teachable Machine, ML4Kids, Skill Blueprints, Cognimates, Scratch and the Flow Editor, may enable students to train simple speech, intent classification or agent models, but are not intended to teach conversational AI concepts and enable flexible, extendable agent development [34, 91, 9, 51, 50, 157]. ConvoBlocks was designed to fill this gap and empower students to develop their own complex agents while learning about the capabilities, limitations and implications of conversational AI through constructionism and associated curriculum [185, 189, 186, 191].

In Chapter 4, I analyze a number of these development platforms, including ConvoBlocks, with respect to the conversational agent concepts they can teach. As described in the chapter, I select specific platforms to address a range of programming abilities—from no experience to advanced experience. These platforms, from least required experience to most, are Zhorai [99], ConvoBlocks [185, 191, 193], Convo [190, 215], the Alexa Developer Console [12] and Google Dialogflow CX [62].

Pedagogical Conversational Agents

Other related work includes pedagogical conversational agents, which typically teach students through interactive, natural language conversation and can be found in the K-12 context [81, 99]. For example, agents might explain virology while touring students through virtual museums [140], assess reading comprehension by engaging as reading partners [211], or encourage students to ask deeper questions through offering question starters [3].

Researchers have investigated which aspects of pedagogical conversational agents help engage students in the learning process. For example, one study found agents could facilitate collaborative learning by providing suggestions and gaze feedback [71]. Other studies have shown how students' perception of their relationship with embodied pedagogical conversational agents can affect their learning [89]. For instance, one study found social agents with high rapport with students were able to improve language learning [88]. Another study found increasing embodied agents' speech mimicry and backstory resulted in increased student engagement, enjoyment, feelings of social relationship and learning success [87]. Thus, it is important to consider students' perceptions of and relationships with pedagogical agents during their learning.

One well-researched and effective learning strategy includes the learning-by-teaching paradigm, which a number of pedagogical agents use to engage students [25, 96]. For example, in the "Betty's Brain" conversational agent environment, students read about a science topic, teach what they learned to an agent, and have the agent take a quiz [26]. Another example of this includes "Curiosity Notebook", in which students choose to teach a conversational robot about animals, rocks or paintings [96]. Using a similar method, the Zhorai platform (which is analyzed in Chapter 4) teaches children science as well as machine learning concepts [99]. Students engage with Zhorai, a conversational agent, teaching it about animals, while Zhorai tries to classify the animals into different ecosystems based on the given information.

These types of K-12 pedagogical agents are not necessarily intended to teach conversational agent concepts. However, students may learn fundamental agent concepts merely by interacting with them. For instance, in Chapter 4, I describe how students can learn about how agents can have various synthesized voices, have a particular conversational pattern and reuse contextual information. Simple interactions with agents such as these, as well as the more complex interactions with conversational agent development platforms described previously, can both provide useful learning opportunities in the classroom.

Perceptions of Conversational Agents

Despite commercial conversational agents being used by children in the classroom as well as homes, few studies have investigated children's perceptions of them [148, 109, 214, 58]. Understanding children's perceptions and feelings towards such agents can likely help educators better facilitate student learning, policy makers ensure safety measures for child-agent interactions are developed, and children themselves better calibrate their understanding and trust of such agents [193, 58, 182, 89]. A recent review of child-agent human-computer interaction research identified a need to better understand children's perceptions of and expectations for their conversational partners, or their "partner models" [58, 49].

Partner models can be described in terms of three main dimensions: (1) competence and dependability, (2) human-likeness, and (3) cognitive flexibility [42, 49]. People's partner models can significantly affect how people interact with agents. For instance, researchers have found that people make different language choices depending on their initial expectations of partner models [49, 42]. Designing agents that produce partner models that align with the capabilities of the agent (e.g., producing a partner model of perceived limited flexibility, if the agent is truly limited in flexibility), could help minimize user frustrations and ease conversation [49]. However, a deep understanding of conversational agent users' partner models—and especially children's partner models—is not reflected in the literature [49, 58].

Certain studies have investigated children's general perceptions of conversational agents. For instance, one study found that the majority of 5-6 year old children considered agents to be friendly, alive, trustworthy, safe, funny, and intelligent [102]. Another study investigated 3-10 year old children's perceptions, and found that children had different perceptions of agents' intelligence depending on the modality of interaction with conversational agent. My past research found students perceived agents to be more intelligent and felt closer to them after learning to program them [193].

Other researchers focus on perceptions of robotic or embodied agents. For instance one study investigated how sociable, mutual-liking, attractive, human, close, and intelligent children 10-12 years old perceived robots to be, finding evidence for learning benefits with anthropomorphized robots [109]. Another study investigated students' perceptions of robots through programming and societal impact activities, finding that children who engaged in the societal impact activities found the robots less trustworthy than those in other groups [47]. Still others have investigated children's relationships with robots. For instance, Boulicault et al. discuss how child-robot relationships are inevitably inauthentic, similar to relationships with imaginary friends or toys, and should be designed with stakeholders in mind, including parents, teachers and children themselves [29]. Another study found researchers can model child-robot relationships through a number of measurement scales, including the Inclusion of Other in Self scale [205]. Using this scale, DiPaola found no significant differences in children's perceived closeness with robots after engaging with them for eight weeks [47]. These results are unlike results from my past study, in which I found children's perceived closeness increased after engaging with and programming agents [193]. Thus, there is a need for further investigation of the differences between perceptions of purely conversational agents and embodied robots, as well as how time scales affect these perceptions.

To my knowledge, there are no studies to date that specifically investigate children's partner models of robots or agents. This is despite how understanding people's partner models can improve agent design. For instance, if designers model agents to ensure users' expectations are met or addressed, users are less likely to encounter frustrations [49]. Thus, by studying people's agent partner models, researchers can develop better design recommendations for agents.

Since different people can have different prior biases and understanding of agents, people from different places and of different ages may have different partner models of agents. Thus, when designing agents, it is likely important to understand different people's partner models. In Chapter 3, I investigate the partner models of children and parents from non-WEIRD and WEIRD countries, and develop corresponding conversational agent usability design guidelines. For example, I found that children from WEIRD countries thought agents were more flexible than those from non-WEIRD countries did prior to the educational activities. Based on this result, agents intended for children from WEIRD countries may need to indicate their inflexibility, or agents intended for children from non-WEIRD countries may need to indicate their flexibility, depending on the actual flexibility of the agent. The guidelines in Chapter 3 aim to empower learners and developers to create effective agents, particularly through conveying appropriate agent partner models.

1.3 Summary of Contributions

The main contributions of this dissertation fall under the following categories.

1. Prominent, comprehensive conversational agent design guidelines to teach to K-12 students (RQ1.0.1, Chapter 2 and 3):
    - A comprehensive list of prominent conversational agent usability design guidelines found in the literature [117], including those developed through investigating children's and parents' trust and partner models during programming and societal impact activities
2. Conversational agent understanding concepts (RQ1.0.2, Chapter 4):
    - Identification of fundamental conversational agent concepts (e.g., training utterances, intents, entities, etc.) through a survey of conversational agent learning tools, including interactive agents and conversational agent development platforms, and associated educational materials
    - Analysis of conversational agent development platforms for people with varied prior knowledge (e.g., Zhorai for novices [99], the MIT App Inventor Conversational AI Interface for beginners [186], Dialogflow CX for expert developers [62], etc.) in terms of which conversational agent concepts they teach
3. Forward-looking design guidelines (RQ1.0.4, Chapter 3):
    - Identification of common themes in student projects and visions for the future of conversational agents from past studies [186, 185, 189, 191, 215] and a final study involving children and parents from WEIRD and non-WEIRD countries
    - Development of relevant forward-looking design guidelines and concepts based on the common themes
4. Teaching guidelines for K-12 conversational agent curricula (RQ1.0.1, RQ1.0.2, RQ1.0.3, Chapter 5)
    - Summary of K-12 teaching guidelines from a systematic literature review of K-12 AI education research, as well as the results of teaching K-12 conversational

agent curricula to children from WEIRD and non-WEIRD countries

5. A pedagogical conversational agent design and understanding framework (Chapter 6):
   - Combination of the findings of all the above research questions, resulting in a pedagogical conversational agent design and understanding framework for students, educators and agent designers

All in all, this dissertation investigates the research questions outlined in Section 1 (RQ1.0.1, RQ1.0.2, RQ1.0.3, and RQ1.0.4) and develops a pedagogical conversational agent design and understanding framework. I developed the framework with a K-12 audience in mind, but it is also likely useful for other audiences. This work is supported by the findings in my past studies with ConvoBlocks, Zhorai, Convo; a past co-design workshop with teachers; and a past literature review of K-12 AI education research, as well as the content provided in the following chapters [185, 191, 193, 189, 137, 99, 190, 215, 184, 188, 214].

Chapter 2

Conversational Agent Design Guidelines

Design creates culture.
Culture shapes values.
Values determine the future.
Design is therefore responsible for
the world our children will live in.
—Robert L. Peters

As discussed in Section 1.2.2, there exists a need for HCI conversational agent pedagogy, including usability design guidelines for those learning to develop agents. This chapter outlines the process of identifying prominent, comprehensive and current usability design guidelines for developing conversational agents. Up until recently, there have been few comprehensive conversational agent design guidelines, due to the nascency of the field and its large recent advances [198, 32, 46, 119, 118]. Within the past few years, however, due to calls to action by researchers and conferences, there has been a surge in the development of such guidelines [119, 118, 57, 129]. To identify the most prominent, comprehensive, recent set of guidelines, I completed a systematic literature review. The final guidelines identified were from Murad et al.'s paper, "Design Guidelines for Hands-Free Speech Interaction" [117].

To develop these guidelines, Murad et al. first explored notable, existing guidelines for GUI design, since they have been established over many years of research. Specifically, they identify Nielsen's [123], Norman's [126] and Shneiderman's [162] guidelines as notable. They then investigate speech human-computer interaction literature for how it addresses the prominent GUI guidelines' recommendations. The authors also identify areas that are unique to speech interaction to develop new speech-specific guidelines. They find that the majority of the GUI guidelines are relevant to speech; however, do not note any papers related to the guideline, "G4: Consistency throughout the Interface" and only note few related to the guideline, "G5: Preventing User Errors". They also identify two areas in speech interaction literature not covered by GUI literature, including "A1: Ensure Transparency/Privacy" and "A2: Considering How Context Affects Speech Interaction" [117].

In Table 2.3, I provide examples for how each of the GUI guidelines can be implemented in conversational agent design, including G4 and G5, based on the authors' discussion, other conversational agent literature, as well as my own research in developing agents and design recommendations [185, 191, 193, 99, 190, 215, 184]. For instance, for "G4: Consistency throughout the Interface" (which the authors did not find any implemented examples in their speech literature review), I found a relevant example suggesting implementing consistent tones of voice, personas, language, as well as other aspects when designing agents [57]. As discussed in the introduction, in later chapters I identify areas not covered by Murad et al.'s guidelines. These include considering people from non-WEIRD countries and people of various ages when developing guidelines [167]. To address this gap, in Chapter 3 I develop additional guidelines to encourage healthy levels of trust and partner model development for various audiences.

2.1 Literature Review

To begin identifying guidelines, I used the snowball surveying technique outlined in Wohlin's framework [208]. I first identified a set of usability guideline papers that met the initial criteria outlined in Table 2.2. I found these papers through a keyword search of relevant search terms, including "conversational agent design guidelines", "voice user interface (VUI) design recommendations", "conversational AI usability heuristics", and "voice assistant design principles". I used Google Scholar to perform the search to reduce bias in favor of specific publishers [208]. The final start set included papers from a variety of conferences and journals, including CHI, HCII, IEEE Pervasive Computing, and MuC [203, 92, 129, 104, 168]. In Table 2.1, I outline the journals and conferences with more

Table 2.1: List of journals and conferences where articles were found through the snowball sampling literature review. The largest number of articles were from the CHI conference, with ten articles. Publications from which there were less than two articles identified fell under the "Other" category. There were 82 articles analyzed in total.

Publication	**Count**
CHI	10
CUI	5
MobileHCI	4
HCII	3
DIS	3
SN Computer Science	2
MuC	2
International Journal of HCI	2
Interacting with Computers	2
IEEE Pervasive Computing	2
IEEE Latin America Trans.	2
ICoRD	2
CSCW	2
Computers in Human Behavior	2
AMCIS	2
Other	37
TOTAL	82

than one article identified in the literature review.

I specifically chose to limit the search to papers discussing voice-based conversational agents, since this dissertation focuses on the unique benefits and risks of voice-based agents. For example, voice-based agents utilize one of the most natural methods of communications humans have used for ages: speech; whereas text-based agents utilize a method of communication recently invented: typing on a technological device. The voice-based method is beneficial, since it is very natural, efficient and enjoyable for humans to learn and use [190, 151]. However, due to humans' long history of voice-based interactions, voice-based agents may be especially personified. This presents a risk, as researchers have found correlations between increased personification of agents and increased trust [160, 193]. Furthermore, researchers have linked increased trust of sources of information to increased mis- or dis-information spread [210, 159]. Thus, it is particularly important to mindfully consider how to design voice-based conversational agents. This is especially

Table 2.2: Criteria for inclusion in the snowball sampling literature review.

Criterion	**Further information**
Voice-based	Conversational agents under investigation must not be specifically text-based (e.g., online text chatbots) or multi-modal (e.g., embodied robotic agents), and include at least some speech component.
General-purpose agents	Conversational agents under investigation must not be developed for a specific user-group or purpose (e.g., specific to health-care, pedagogy, automobiles, etc.).
General guidelines	Conversational agent guidelines must not be specific to only one aspect of agent design (e.g., specific only to naturalness or trust).
List specific evaluative usability heuristics or recommendations	Conversational agent guidelines must be listed and must evaluate potential agents in terms of usability (e.g., the paper cannot just describe a model of agent language or only list challenges of creating agents, but must evaluate agent usability).
From 2017 onwards	Paper must be published in 2017 or later due to large advances in natural language processing and speech recognition in 2017 (e.g., transformer models [198]).
English	Paper must not be in a language other than English (although the agent being evaluated may be speak in another language).
Peer reviewed	Paper must be published in a peer-reviewed journal or conference

true considering reports indicating an exponential rise in the use of voice-based agents [163, 217].

Through backwards and forwards snowball sampling, I identified 82 candidates through their titles and abstracts as potentially falling under the outlined criteria. Table 2.1 shows the journals and conferences where the majority of the candidates were published. After reviewing the papers further, I determined 16 of the candidates fell entirely under the criteria. Of the 16 candidates, five were survey papers that aggregated information from many sources to determine overarching guidelines. The final, most prominent paper with aggregated guidelines was Murad et al.'s article, "Design guidelines for hands-free speech interaction" [117], which surveyed 21 papers to develop its 12 usability guidelines for speech interfaces. Other researchers had cited this paper at least 66 times at the time of this dissertation's publication. The 12 guidelines from this paper are shown in Table 2.3. In this table, I also describe example opportunities for implementation when developing conversational agents. I identified these opportunities through my literature review (with special attention to Murad et al.'s discussion) and experience with my own conversational agent designs and studies.

Table 2.3: General guidelines for conversational agents from Murad et al.'s article [117] and examples of how to put these guidelines into practice. Note that the guideline titles in this table are directly from Murad et al.'s article, and the opportunities in practice are largely inspired by this article, as well as other literature and opportunities from my conversational agent designs and studies [185, 191, 193, 99, 190, 215, 184].

Design Guideline for Hands-Free Speech Interaction [117]	**Opportunities in practice**
G1: Visibility/Feedback of System Status	• Indicate when users can speak to the agent. • Indicate the agent's capabilities. • Provide feedback on the agent's interpretations of users' speech and how users can speak differently to improve recognition.
G2: Mapping Between System and Real World	• Provide example conversation schema and scaffolding. • Since users mimic agents' speech patterns, develop agents that speak in the way users should speak for best speech recognition (e.g., using complete sentences [175])
G3: User Control and Freedom	• Provide sufficient time for users to speak to the agent and understand its speech. • Allow users to interrupt the agent (see G7). • Allow for flexibility in the flow of the conversation; for example, by allowing users to provide additional information at various times in the conversation (see G7).
G4: Consistency throughout the Interface	• Use consistent conventions (e.g., terminologies, tone of voice, language) when designing conversation and conversational flow [57].
G5: Preventing User Errors	• Enable agents to understand wide-ranging diction, idioms, and phrases (see G7). • Confirm choices when users make consequential decisions to ensure statements were understood correctly (see G9). (Implicit confirmations may be used to address G8.)
G6: Recognition Rather than Recall	• Provide information to users on how to complete tasks and structure responses, rather than forcing them to guess.
G7: Flexibility and Efficiency of Use	• Ask users open-ended questions, rather than yes or no questions (e.g., "Where would you like to go?" vs. "Do you know where you'd like to go?" [175]). • Allow users to interrupt the agent (see G3). • Allow for flexibility in the flow of the conversation; for example, by allowing users to provide additional information at various times in the conversation (see G3). • Enable agents to understand wide-ranging diction, idioms, and phrases (see G5). • Minimize or scaffold the amount of information and options given to the user (see G8).

G8: Minimalism in Design and Dialogue	• Use implicit confirmations in feedback statements rather than repetitive, explicit confirmations when possible. • Minimize or scaffold the amount of information and options given to the user (see G7).
G9: Allowing Users to Recognize and Recover from Errors	• Indicate when speech recognition is uncertain, and enable users to repeat or rephrase their responses. • Enable agents to recognize when users want to edit misunderstood responses. (This may be when users repeat their response, when they speak more loudly, or explicitly ask to edit.) • Allow users to undo actions or go back to previous menus. • Confirm choices when users make consequential decisions to ensure statements were understood correctly (see G5). (Implicit confirmations may be used to address G8.)
G10: Providing Help and Documentation	• Provide interactive tutorials. • Provide scaffolded help contextually throughout interactions.
A1: Ensure Transparency/Privacy	• Provide information on what data is being collected. • Recognize quiet or whispered speech (or consider silent speech recognition, like lip reading [133]) to increase privacy in public spaces (see A2).
A2: Considering How Context Affects Speech Interaction	• Enable various devices or input modalities to increase social appropriateness of interacting with agents in different spaces (e.g., speaking vs. inputting text in a public space, speaking to a smart speaker vs. a phone while in conversation with others). • Recognize quiet or whispered speech (or consider silent speech recognition, like lip reading [133]) to increase social acceptability in public spaces (see A1).

2.2 Design Recommendations' Connection to the Framework

In later chapters, I structure the HCI portion of the K-12 pedagogical framework based on the usability guidelines in Table 2.3, and analyze how these guidelines are taught through the conversational agent development platforms and tutorials outlined in Chapter 4. I also contribute additional guidelines based on a lack of guidelines in the literature considering people of different programming backgrounds, generations, cultures and gender identities [117, 167]. Specifically, I investigate differences in learning and perceptions of conversational agents of children and parents from non-WEIRD and WEIRD countries, and develop guidelines for how to address these differences. The investigation and associated guidelines are outlined in Chapter 3. Finally, I develop guidelines for teaching such design recommendations to K-12 students in Chapter 5.

Chapter 3

Conversational Agent Development and Perceptions Study

How will [conversational agents'] ubiquity and
integration in our daily lives change how we live?
Who will be most affected by the decisions of agent owners?
—Ruane et al. [150]

As discussed in Chapter 1, analysing people's perceptions of conversational agents is important in order to both better develop effective conversational agents and better teach people about conversational agents. This is especially important for *conversational* technology since it is inherently personified, and people naturally build relationships with it [160]. As people build trust with agents, agents become uniquely positioned to both engage in effective teaching (as trust is important in teacher-student relationships [113, 67]), as well as spread misinformation (see Figure 3-1).

In this Chapter, I outline a study with ConvoBlocks, in which three undergraduate student researchers, Nguyen, Tian and Kelleher, and I investigate how children and parents from eight different countries perceived and trusted agents during an educational activity. The purpose of the study was to determine whether educational activities—in this case, programming and societal impact activities—can change people's perceptions of agents. As discussed in Section 1.2.3, by better understanding how different people perceive agents, we can learn to better develop agents and educational activities, thereby better equipping people for a conversational-agent-filled-world.

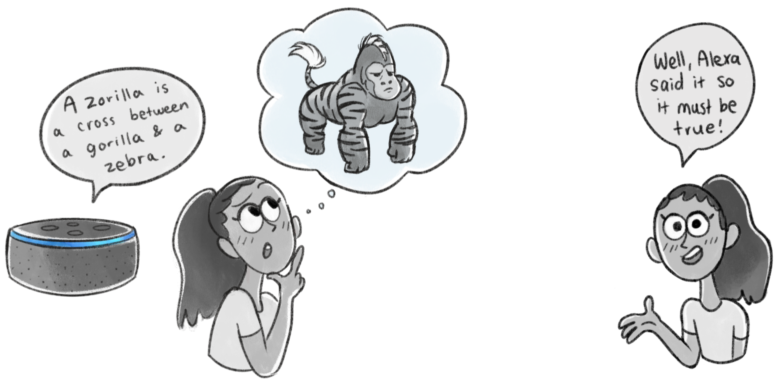


Figure 3-1: A depiction of overtrust of Alexa [196]. (A zorilla is a skunk-like animal largely found in Africa.) Although this is a trivial example, overtrust can have serious implications for misinformation spread.

3.0.1 Study Novelty

Despite a vast majority of the world's population being from locations that are not Western, educated, industrialized, rich and democratic (WEIRD), much of the non-WEIRD population has been neglected in the literature—including HCI literature [167]. Furthermore, researchers have called for more conversational agent-related studies with participants from diverse locations and backgrounds [160]. This chapter expands on my previous research on children's perceptions of conversational agents, involving similar agent-development workshops [191, 193, 186]. These past studies, however, only focused on WEIRD perspectives [167], and children's perspectives; whereas this study includes non-WEIRD and parents' perspectives, in addition to WEIRD and children's perspectives. These past studies also only investigated general aspects of trust and perceptions, instead of engaging with specific theoretical models of trust and partner models, as described in the next sections.

3.0.2 Modeling Partner Models

As discussed in Section 1.2.3, the partner model is a useful construct when investigating people's expectations for conversational partners, including agents. When user expectations of conversational partners match reality, users are better able to understand their partners' choices, better able to predict their partners' future actions, and less likely to be frustrated by their conversational partner. Partner models can also influence how users interact with agents; for instance, users might align their syntax with their model

Table 3.1: Partner model dimensions from Doyle et al.'s model, and related terminology used in surveys to investigate participants' perspectives of Alexa [49]. This terminology is largely from Doyle et al.'s psycholexical analysis and provides further insight and detail into the dimensions [49].

Dimension	**Survey terminology**
Competence and dependability	• Unreliable vs. dependable • Competent vs. incompetent
Human-likeness	• Human-like vs. machine-like • Warm vs. cold • Friend vs. co-worker • Authority figure vs. peer
Cognitive flexibility	• Interactive vs. start-stop • Flexible vs. inflexible

of their partner's syntax [41] or make assumptions based on their model of their partner's way of speech [43]. In this dissertation, I adopt Doyle et al.'s model, which is the first to identify key dimensions of partner models in conversational agent interaction. The key dimensions include: (1) competence and dependability, (2) human-likeness, and (3) cognitive flexibility [42, 49], as shown in Figure 3-2. We investigate participants' partner models of Alexa in terms of these dimensions, using the terminology outlined in Table 3.1 to uncover specific aspects of their perceptions.

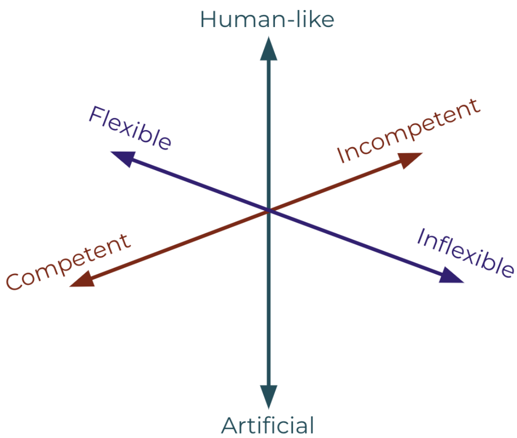


Figure 3-2: The key dimensions of partner models, as outlined by Doyle et al. [49].

Furthermore, people's partner models can influence their sense of trust of agents. For instance, Cowan et al. found that people trusted conversational partners differently to give correct answers depending on their partner models [42]. This reveals a core difference between how users interact with voice user interfaces and other smart technologies. Voice user interfaces are inherently human-like, and therefore likely to be immediately anthropomorphized and interacted with socially [160]. This brings additional challenges. As Turkle once put it, "A virtual assistant or chatbot that offers friendship reduces a person to lines of code", or in other words, relational chatbots and their designs influence how humans interact with them, as well as with other humans [181].

Researchers are beginning to investigate this phenomenon and how agents' anthropomorphized designs affect how users interact with them. For instance, Seymour and Van Kleek found interactions with conversational agents can be modelled using the same metrics as social relationships. They also found a correlation between measured relationship development with agents and trust of agents [160]. This correlation is especially concerning (and much less researched [52, 193]) when considering children and their vulnerability. This study investigates children's relationships with conversational agents in in terms of partner models, and how researchers can develop pedagogical interventions and conversational agent designs to encourage balanced calibration of trust and perceptions of agents.

In this study, we found partner models changed differently for different subsets of the participants through the activities. For instance, after the programming activities, participants from non-WEIRD countries felt agents were more competent, more dependable and more of an authority figure than those from WEIRD countries did. Another example includes how children felt Alexa was warmer and more human-like than parents did throughout the activities. Thus, I recommend designing conversational agent personas to foster partner models appropriate for the particular task (e.g., social vs. co-working agent) and end users. For instance, for end users from non-WEIRD countries, developers may want to focus on creating competent personas (assuming the agent is truly competent). For parent end users, developers may want focus on creating warm and human-like personas. This also aligns with how we found parents described their ideal agent designs with a focus on human-likeness. These traits are also linked to trust of agents, as discussed in the next section.

3.0.3 Modeling Trust

There are many different methods of defining, modeling and assessing trust. For instance, sociology researchers have defined it with respect to people's interests not being hurt by another party; philosophy researchers with respect to personal, moral relationships and risky actions; and computing and networking researchers with respect to how likely it is for an entity to behave reliably [37]. One well-recognized model of trust involves four characteristics: (1) Competence, (2) Benevolence, (3) Integrity and (4) Predictability. The authors of this model, McKnight and Chervany, developed it through combining multiple different disciplines' constructs of trust to create a "genericized" model for use in many research domains [107]. Indeed, researchers have referenced this model in thousands of different works, including investigating trust with respect to agents [23, 8, 103], website design [132, 139, 56], and even virtual reality interactions [164, 54, 55]. I also adopt McKnight and Chervany's model of trust to investigate children and parents' perceptions of conversational agents.

Alongside Nguyen, Kelleher and Tian, I analyze participants' trust in two main ways. The first is through a thematic analysis of long-answer responses to a question about whether participants felt their trust changed through the workshops, and why. The second is through analysing the responses to Likert scale questions, which ask participants to rate their general trust of agents, and their specific trust in the correctness of agents' information. We are particularly interested in trust of information correctness, as researchers have linked trust to the spread of misinformation [210, 159]. In the thematic analysis, we developed themes inductively (as in Braun et al.'s description of this method [30]) and later categorized relevant themes into McKnight and Chervany's four trust characteristics [107].

We found that participants focused most on the competence, then predictability and then integrity aspects of trust with regards to agents. More specifically, participants frequently mentioned where agents obtained their information, what they do with the information they are given and how they were programmed when discussing their trust of agents. Thus, I recommend informing end users about these topics through agent design and educational interventions to increase transparency and allow end users to calibrate their trust accordingly.

3.0.4 Computational Action and Technology Democratization

As described in Section 1.2.1, developing curricula to empower students to develop technology to affect their communities—or engaging students in "computational action"—can increase their engagement, and provide them with a more inclusive learning environment [176, 177]. In the curriculum we developed for this study, we teach students to develop their own conversational agents and engage them in designing their ideal "future worlds" (whether that includes conversational agent related components or not). Through these activities, we investigated how different people's self-efficacy and identities as programmers changed. For example, we found participants' confidence in feeling like a programmer and in creating their own technology significantly increased through the programming activity. However, participants' confidence in being able to make an impact in their community or the world using technology only significantly increased after the societal impact activity. We also investigated which conversational agent concepts were most challenging for participants. These included Training, Terminology and Turn-taking. Based on these results, I present teaching recommendations for K-12 conversational agent curricula, including teaching both programming and development activities to facilitate computational action, emphasizing challenging concepts when teaching, and others.

By making technology understandable and accessible to everyone—or democratizing technology—more diverse voices can contribute to how technology is built and which problems it addresses. Many AI education initiatives and workshops aim to democratize technology through education [186, 215, 134, 100, 178]. Since not everyone has access to such initiatives and workshops and the ability to create their own agents, however, we also investigated how various groups (children and parents from non-WEIRD and WEIRD countries) generally envisioned their ideal conversational agents. That way, developers who already have the knowledge and skills to affect change in the world of technology can take diverse people's opinions into account when creating new agents. For instance, developers may want to design more task-oriented agents, since participants generally described their ideal agents this way. They also described their agents with a balance between artificiality and human-likeness, and frequently described agent features such as useful, common features; user-oriented features; fun features; and emotionally intelligent features.

3.0.5 Research Questions

The purpose of this Chapter is to develop design recommendations to complement the prominent, general usability recommendations outlined in Chapter 2. The recommendations from Chapter 2 are based on research largely from a WEIRD context [117, 124, 161] and aimed at adults with deep prior knowledge of computer science. The authors of the recommendations note that their exploration of established guidelines provides a foundation for future conversational agent usability research to build upon [117]. The recommendations developed in this chapter extend these recommendations through addressing a wider audience—including those from WEIRD and non-WEIRD countries and those from different generations—and considering how agent programming and societal impact activities can affect partner models and trust of agents. They also focus on more than just conversational agent usability, and include recommendations for conversational agent pedagogy and recommendations for how to align conversational agent development with different user groups' ideal future agents. In later chapters, I relate and adapt the recommendations I develop in this chapter (as well as those from Chapter 2) to a K-12 context, which informs the K-12 pedagogical conversational agent design and understanding framework.

In this chapter, I aim to answer the following research questions:

RQ3.0.1: How do people of various backgrounds (WEIRD and non-WEIRD, as well as different generations) perceive Alexa with respect to partner models [49] and trust before, during and after conversational agent programming and societal impact activities?

- Furthermore, how might these perceptions inform design recommendations for agents and educational interventions?

RQ3.0.2: What do people of various backgrounds find most difficult when learning about and developing conversational agents?

- Furthermore, how might this inform teaching guidelines for conversational AI?

RQ3.0.3: How do people of various backgrounds feel in terms of self-efficacy and identity as being programmers through conversational agent programming and societal impact workshops?

- Furthermore, how might this inform teaching guidelines for conversational AI?

RQ3.0.4: How do people of various backgrounds envision the future of conversational agents?

- Furthermore, how might this inform design recommendations for future conversational agents for these audiences?

The results of the investigation into these research questions led to recommendations for conversational agent usability (DR-Us), conversational agent pedagogy (DR-Ps), and how to align conversational agent development with users' ideal future conversational agents (DR-Fs). Table 3.2 summarizes the design recommendations. Each of these recommendations contribute to the development of the K-12 pedagogical conversational agent design and understanding framework, in turn helping educators create pedagogy to empower others to better understand and develop conversational agents.

Table 3.2: The conversational agent and pedagogy design recommendations developed based on the results of the study.

Purpose	Recommendation
DR-U: To help developers create conversational agents with increased usability and learnability	**DR-U1:** Inform users about trustworthiness **DR-U2:** Design conversational agent personas to foster appropriate partner models
DR-P: To help educators create effective conversational agent pedagogy	**DR-P1:** Encourage trust of pedagogical agents (to the extent of their trustworthiness) to facilitate learning from them **DR-P2:** Engage students in activities that will reinforce the concepts being taught with respect to their conversational agent partner models **DR-P3:** Teach visual programming to empower nearly anyone to program conversational agents **DR-P4:** Emphasize concepts that are challenging for particular audiences **DR-P5:** Include both societal impact and conversational agent development activities to facilitate computational action **DR-P6:** Encourage and provide consistent programming opportunities to underrepresented minorities **DR-P7:** Supplement conversational agent development activities with additional agent engagement, AI learning and programming activities

DR-F: To empower current developers with a greater understanding of how people—other than themselves—may want technology to be developed, and thereby aid in technology democratization	**DR-F1:** Design with more task-orientation in general, while considering the end user audience **DR-F2:** Balance personification and artificiality in agent design while considering the end user audience **DR-F3:** Focus development on useful, common features; user-orientation; enjoyable interactions; and emotional intelligence, while emphasizing certain aspects depending on the end user audience

3.1 The ConvoBlocks Interface

This section describes ConvoBlocks (aka the MIT App Inventor Conversational Agent Platform), which is the platform participants used to develop conversational agents during the study. Similar information can be found in my master's thesis [186] and related papers [185, 191, 193].

ConvoBlocks is a block-based programming tool to teach young people conversational AI while empowering them to create socially useful conversational agents. I have used this system in two previous studies investigating the usability and effectiveness of the interface, how to best teach conversational AI concepts to K-12 students, and how students' perceptions of conversational agents change before and after programming conversational agents [185, 186, 193, 191]. To illustrate the system, I present a fictional, motivational scenario in which a student, Sheila, creates an app and conversational agent with ConvoBlocks (as presented in my paper [185]). Although the scenario is fictional, a version of *Sheila's Storybook App* was implemented using the interface and presented to students as an example during K-12 workshops. A diagram showing the user workflow is shown in Figure 3-3.

Motivational Scenario

Sheila, a seventh grade student, loves stories. When she was younger, she imagined jumping into the pages of her storybook and interacting with the characters. During a computer lesson, she heard about MIT App Inventor's conversational AI interface, ConvoBlocks, and had a brilliant idea: to create a talking storybook. Sheila would create the storybook app using ConvoBlocks, run it on her tablet, and enable conversation using the Alexa app. The storybook would have the following main features:

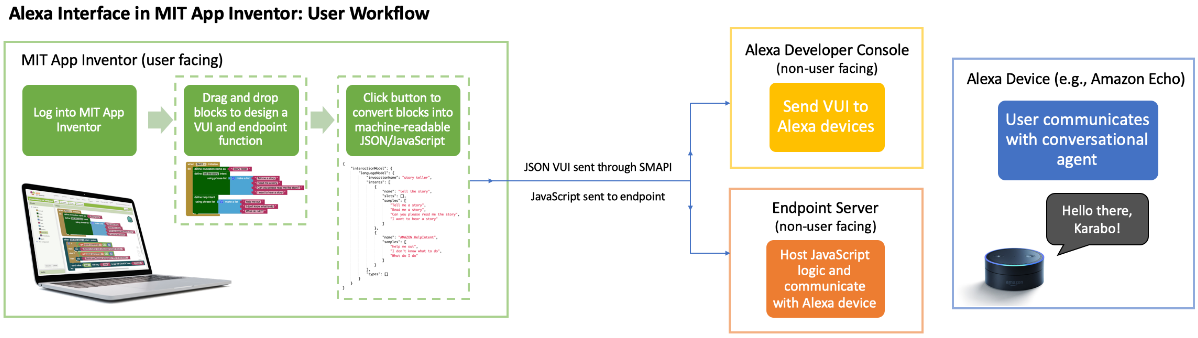


Figure 3-3: User workflow to create a conversational AI agent. The user first implements the Voice User Interface (VUI) and endpoint function using a block-based interface. The blocks are converted to JSON and JavaScript, which define the agent's functionality on Alexa devices [185]. See Section 3.1 for more details about the implementation.

- You could swipe through virtual pages of the storybook while reading and viewing illustrations on-screen
- You could ask Alexa about the characters, setting, and narrative (e.g., Figure 3-4)
- You could ask Alexa to read you the story, and as Alexa reads, the sentence on the app's page would be highlighted
- You could have virtual conversations with the storybook characters, and when you ask a character a question, a response would be automatically generated

Figure 3-4: Speaking with Alexa contextually with Sheila's storybook [195, 185].

To implement the storybook app, Sheila first uploaded illustrations to MIT App Inventor and implemented page-flipping functionality by creating *events* in the block-based interface. When the "next" button was pressed, a counter would increase, and the next illustration would pop up on screen. The opposite event would occur when the "previous" button was pressed. After creating the mobile app (as shown in Figure 3-5), Sheila moved onto the conversational AI portion of the app with MIT App Inventor's ConvoBlocks.

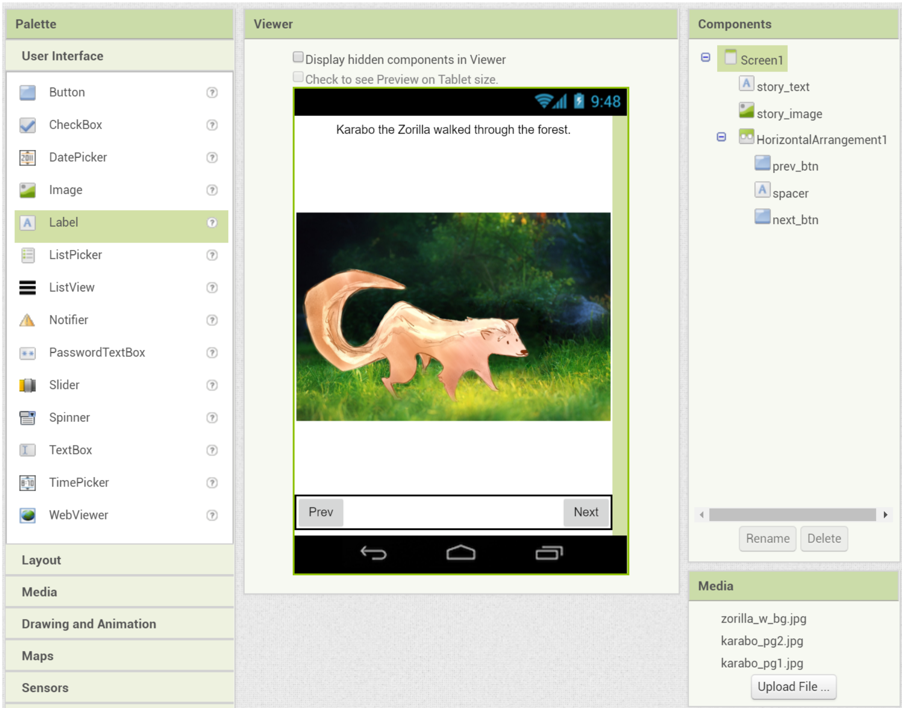


Figure 3-5: The storybook mobile app being developed on the MIT App Inventor website [185].

Sheila added a new Alexa Skill to the project by clicking the *Add Skill* button. This brought her to a page where she could drag-and-drop and connect blocks together to create a conversational agent, as in Figure 3-6. First, she dragged out a *define intent* block so that the Voice User Interface (VUI) would recognize when someone said, "Tell me a story". She wanted the agent to respond by telling a story about zorillas (a little-known animal that Sheila absolutely *loved!*), so she also dragged out a *when intent spoken* block. She connected the *when intent spoken* block to a *say* block containing a *text* block with the first line of the story.

Sheila also wanted people to be able to speak with the main character, Karabo the Zorilla. She didn't want to write out all the possible answers to people's questions though (that would take *forever*), so she decided to use a *generate text* block to generate Karabo's answers instead. According to her seventh grade teacher, this block used *machine learning* to generate sentences sounding kind of like the stories in the block's drop-down menu. Sheila imagined Karabo speaking kind of like Dr. Seuss, so she chose that one from the menu.

After adding some additional functionality using blocks, as shown in Figure 3-6, Sheila sent the Alexa Skill and mobile app to her cousin Jaidon. Jaidon downloaded the app and started flipping through the pages, talking to the storybook as he went along. Jaidon was thrilled that he could listen to Alexa read him the story, especially since he didn't know how to read himself. He also had a blast asking Karabo questions and hearing the infinite different ways Karabo would respond (despite the sentences not always being perfectly logical). He laughed when Karabo said, "Little cat in the hat, I cannot eat them in the snow". It sounded quite like one of his favorite Seuss stories. Hearing Jaidon's laughter meant the world to Sheila. In her mind, an app that allowed her to connect with her cousin thousands of miles away was a huge success.

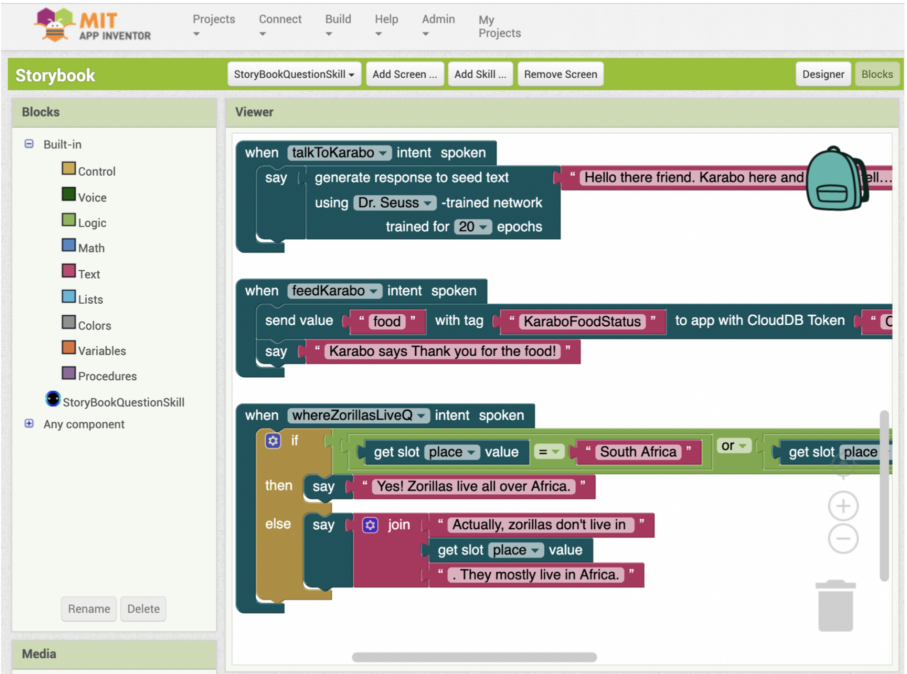


Figure 3-6: Sheila's storybook endpoint function blocks in MIT App Inventor's ConvoBlocks. Notice the *when intent spoken*, *say*, and *generate text* blocks [185].

Although the above story is fictional, students have created many similar agents using ConvoBlocks, including sign-language teaching agents, agents to help diagnose illnesses, and cookbook agents [185, 191, 193], as shown in Figure 3-7 and Figure 4-14.

Figure 3-7: A visualization of a conversation turn with a cookbook agent. The recipe step in this turn also appears on-screen with an image in an associated app.

Design and Technical Implementation

As described in my paper [185], the main goals of the conversational AI interface's design were to empower students with little or no programming experience to **(1) learn computational thinking skills**, **(2) learn conversational AI concepts**, and **(3) develop conversational AI applications**. Figure 3-8 illustrates examples of how the interface implements these three goals, including how code blocks, like the "generate text" block, can teach conversational AI concepts. I provide more details about how the MIT App Inventor team and I developed the interface in my master's thesis and related papers [185, 186, 193].

3.2 Procedure

The following section describes the workshop, data collection and data analysis procedures.

3.2.1 Workshop Outline

We designed ConvoBlocks workshops for students aged 11-18 and their parents to learn about conversational agents, how to program them, and what it means to have them in the world. In the workshops, students and their parents completed programming and societal impact curriculum over two days. Each day involved 3.5 hours of Zoom interaction.

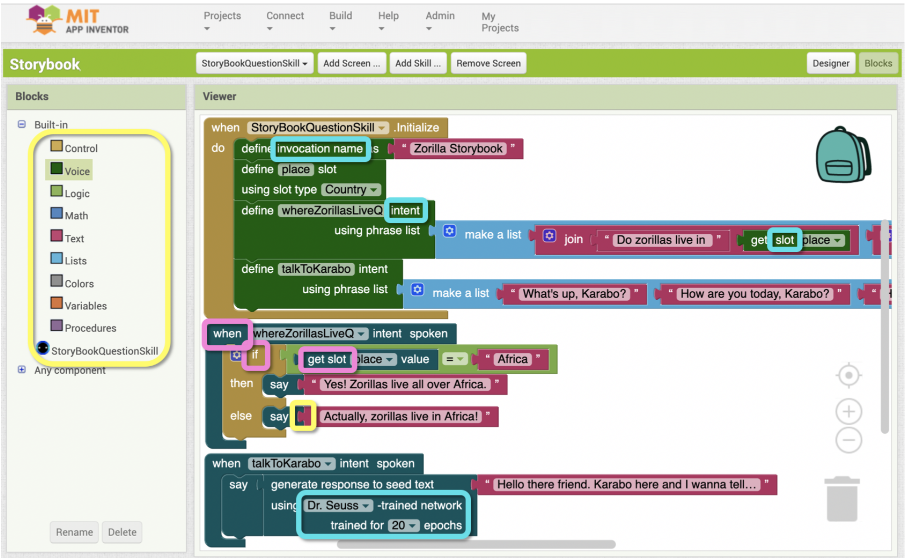


Figure 3-8: The ConvoBlocks interface highlighted according to design goals. *Pink:* The pink boxes highlight the interface's attention to computational thinking skills, including events (*when*), conditionals (*if*), and data (*get slot*). *Blue:* The blue boxes highlight conversational AI concepts, including *invocation name*, *intent* and *slot* (i.e., entity), as well as a "generate text" block containing LSTM neural networks. *Yellow:* The yellow boxes illustrate the learnability of the interface. The leftmost highlight shows the "drawers" where blocks can be easily dragged-and-dropped into the interface, and the smaller yellow box shows how block-based coding can prevent syntax errors, as only certain blocks (e.g., "say" and "text" blocks) can be connected to each other [185].

Day 1

At the start of the first day, we provided participants with a pre-survey (as described in Section 3.2.2). Next, we led participants through a tutorial, introducing them to ConvoBlocks and key conversational AI vocabulary, like "intents" and "training". The agent developed in this tutorial could respond with basic information about carbon footprints when asked. Throughout the day, students could either follow along with the group tutorials or go ahead and complete extra tutorials.

Next, participants engaged in an ideation session in which they envisioned the future of conversational agents, including answering prompts about what their ideal agent might look like, sound like, say, do and discuss. After this session, participants learned to build another agent. This agent demonstrated the fundamental conversational AI concept of entity extraction (as described in Chapter 4), enabling the agent to compute participants' carbon footprint based on the information they gave about their commuting habits.

Towards the end of the programming activities, we demonstrated a working version of the agent from the final tutorial. Only students who had gone ahead were able to complete this tutorial in the allotted time. This agent engaged users in back-and-forth conversations, collected information about users' monthly habits, and ultimately computed their overall yearly carbon footprints. We encouraged students to try developing this agent on their own and expand its functionalities. The first day concluded with participants filling out a mid-survey, where they revisited their choices and responses in the pre-survey, assessing whether their opinion had changed.

Day 2

The second day involved three rounds of societal impact activities. First, participants formed teams of approximately three children and their parents. In the first round, we gave a presentation about sustainability and human mindsets. Afterwards, teams discussed the presentation and how it related to conversational agents and human identities. They then identified mindset changes people could make to foster earth's sustainability.

In the second round, we gave a presentation on the current condition of the planet (e.g., air pollution, coral reef destruction, energy use). Teams reconvened to discuss their ideas for the three most important environmental and technological changes people could make to foster earth's sustainability. Students and parents shared the knowledge they gained from their experience with conversational AI and discussed how AI could help with the issues presented. Finally, each team presented their ideas to the group. Throughout the activities, participants could develop conversational agents which could either help address the challenges presented, or help them present their ideas. Afterwards, students completed a final post-survey, reflecting on how their opinions about themselves, conversational agents, and their partner models may have changed.

3.2.2 Data Collection

We collected all data for the analysis during the two, two-day workshops. One of the workshops had participants from WEIRD countries and the other had participants from non-WEIRD countries. We collected the data through ideation sessions using a whiteboard sharing website called Miro [110] and through surveys. Figure 3-9 shows a portion of an ideation board, where participants outlined their vision for the future of conversational agents, answering prompts about conversational agents' voice, look and functionality. The surveys and boards can be found in full in Appendix B and C. The workshops involved three different surveys: a pre-survey, mid-survey and post-survey. These surveys collected information about participants' demographics; trust of conversational agents; partner models along the different dimensions discussed in Section 3.0.2; and self-efficacy and identities as programmers. The surveys included multiple-choice, 5-point Likert scale and long-answer questions. We told the participants not to look conversational agents up online or discuss with others while completing the surveys.

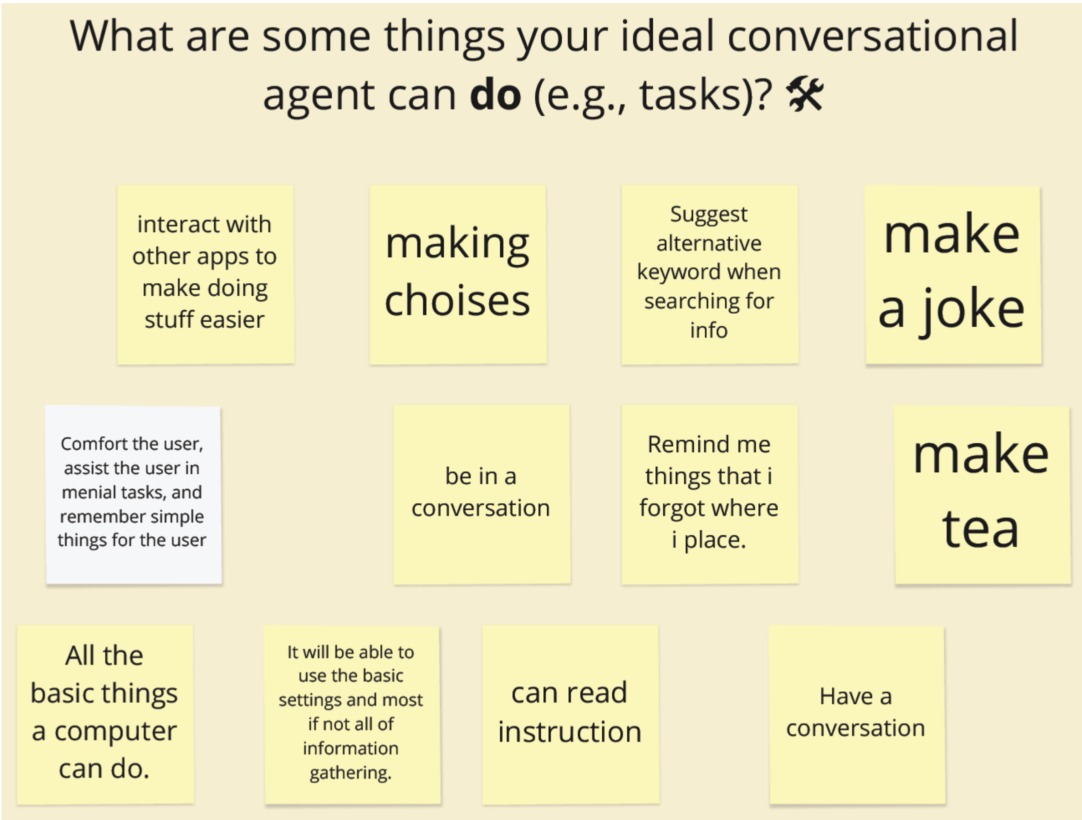


Figure 3-9: A portion of the virtual ideation whiteboards. In this portion, participants from non-WEIRD countries described what their ideal conversational agents would be able to do.

3.2.3 Data Analysis

This section describes how three undergraduate student researchers, Kelleher, Nguyen and Tian, and I completed the quantitative and qualitative analyses.

Quantitative Analysis

The quantitative results from our analysis were found using statistical tests corresponding to the distributions of the data and whether the data were from the same sample of participants (e.g., pre- vs. post-test) or from independent samples of participants (e.g., children vs. parents). The tests and corresponding statistic symbols are outlined in Table 3.3. Note tests with the same samples of participants are indicated with "pre/mid", "pre/post" or "mid/post" (depending on which surveys were compared) in the text. The pre-survey was completed by participants before the entire workshop, the mid-survey was completed after the programming activity and before the societal impact activity, and the post-survey was completed after the entire workshops. Figures in this section display stars (*) to denote significant differences, which correspond to the p-values shown in Table 3.4.

Table 3.3: A list of different statistical tests used in different situations (e.g., with data from the same sample of participants or independent samples, or data from a normal distribution or not), with their corresponding statistic symbols.

	Independent	Normal	Statistic symbol
Independent t-test	✓	✓	t(df)
Mann-Whitney U test	✓		U(df)
Paired t-test		✓	t(df)
Wilcoxon signed-rank test			W(df)

Qualitative Analysis

To identify common characteristics participants desired in conversational agents (as shown in Tables 3.13, 3.14, and 3.15) and themes in long-answer questions (as shown in Table 3.19), three researchers conducted a coding reliability approach to thematic analysis [30]. The researchers performed this analysis using the qualitative data from the workshop surveys completed by the participants (as shown in Appendix B). All of the data were aggregated into one document and partitioned into three sections. Each researcher

Table 3.4: Symbols in figures and their corresponding p-values.

Significance	Symbol
$p \leq .05$	*
$p \leq .01$	**
$p \leq .001$	***

analyzed one section, developing initial ideas for themes. Afterwards, the researchers reconvened to decide on (1) the overarching set of themes for all sections; (2) the set of guidelines and definitions for each theme; and (3) the set of guidelines for highlighting the text.

The researchers completed three rounds of coding so that the Krippendorff's Alpha between all researchers was $\alpha \geq .800$ [16]. For the first round, each researcher coded two out of three sections of the aggregated data (thus, each section was coded by two researchers). The researchers reconvened to compare differences, and the set of definitions and guidelines were updated for coding agreement purposes. For the second round, researchers re-coded their sections based on the updated definitions and guidelines. This occurred twice, until Krippendorff's Alpha was calculated to be $\alpha \geq .800$ between each researcher. The tagged data were then aggregated by union between researchers, and organized with respect to the following categories: WEIRD, non-WEIRD, child and parent.

3.3 Participants

To recruit participants, we sent out interest forms to K-12 education email lists. To be a participant, the only requirements were to have an internet-connected computer and the ability to test Android apps and Alexa-based agents, which could be simulated on the computer, Alexa App or any Alexa device. There were 99 pairs of children and parents from eight different countries who filled out interest forms. In terms of pairs, the top three countries included Indonesia (35 pairs), the US (35 pairs) and the UAE (10 pairs). Overall, there were 47 pairs from WEIRD countries and 60 from non-WEIRD.

Forty-five pairs out of the 99 fully completed the consent and assent forms, allowing them to be a part of the study. Twenty-four of the pairs were from non-WEIRD countries and 21 of them were from WEIRD countries. Within the pairs, there were 45 children and 41 parents (as some parents had multiple children attending) eligible to be a part of the study.

In total, there were 49 participants who completed at least one of the three surveys. Forty-six participants completed the pre-survey, 40 completed the mid-survey, and 35 completed the post-survey. Figure 3-10 shows the countries where participants who completed the pre-survey lived. The children ranged from age 11 to 18. There were few significant differences between those from non-WEIRD and WEIRD countries in terms of demographics questions, as shown by Figure 3-11. The one main difference was how more participants from WEIRD countries spoke English as their first language.

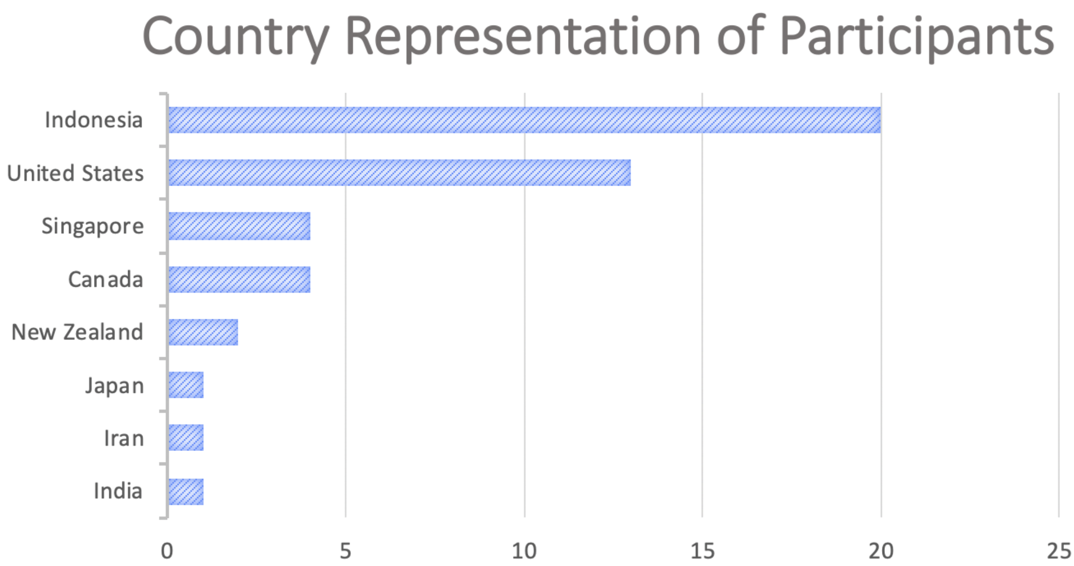


Figure 3-10: Number of participants in the workshop by country. Note that the largest group of participants in the non-WEIRD category was from Indonesia and the largest group in the WEIRD category was from the United States.

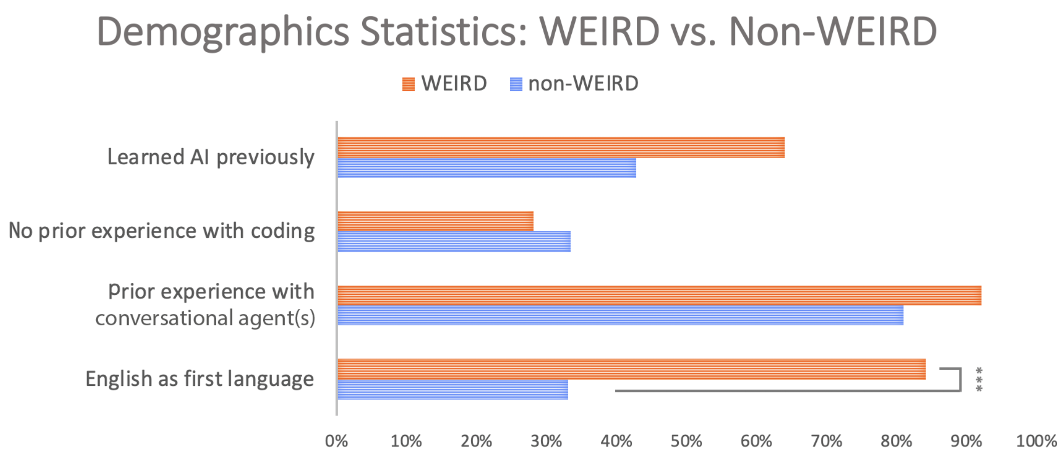


Figure 3-11: Various statistics about participants from non-WEIRD and WEIRD countries. The only significant difference found was how there were more participants from WEIRD countries who spoke English as their first language.

Table 3.5: Percent of long-answer responses indicating a shift in participants' perceptions of agent partner models through the programming activity. The cells in bold indicate the largest percentages for different subsets.

Subset	Changed	Did not change	Ambiguous
Overall participants	**62%**	35%	3%
Non-WEIRD	**76%**	18%	6%
WEIRD	**50%**	**50%**	0%
Children	**67%**	33%	0%
Parents	**54%**	38%	8%

3.4 Results

The following quantitative and qualitative results were found using the methods in Section 3.2.3 and are discussed in Section 3.5.

3.4.1 All Participants

Partner Model

There were few differences between the pre-, mid- and post-surveys in terms of overall participants' partner models. (There were many differences for subsets of the participants, however, as described in following sections.) One outstanding significant difference was how overall participants' feelings towards agents shifted towards more of a friend (than a co-worker) after going through the workshops (pre/post: $\bar{x}$=3.58,3.24; t(32)=2.15; p=.039).

Despite few significant differences in the overall data, 62% of the overall participants indicated they felt their partner models had changed through the programming activity, as shown in Table 3.5. Table 3.5 also shows how the majority of participants from all subsets felt their partner models had changed (except for the subset of those WEIRD countries, in which there was exactly 50% of participants who felt their partner models changed).

In participants' long-answer responses, they most often cited their perceptions of agents' competence and reliability as changing in terms of partner models. The third most-frequently cited aspect was agents' flexibility. Figure 3-12 shows these results, as well as the other components that participants referenced.

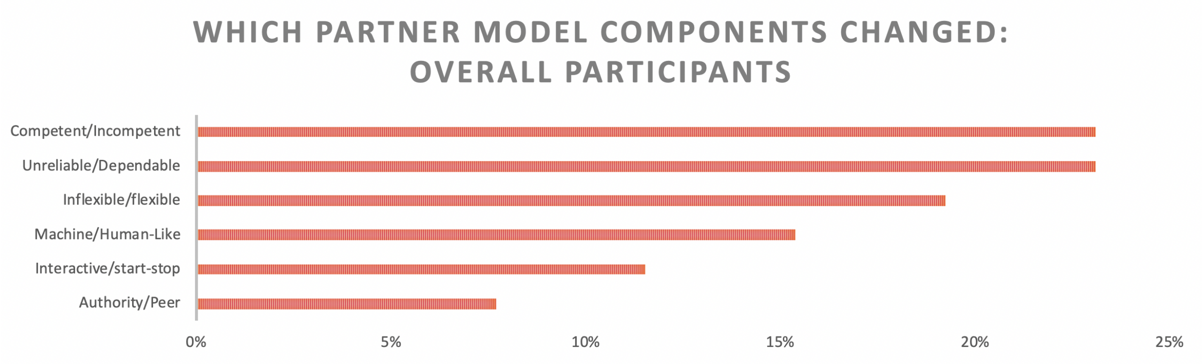


Figure 3-12: The frequency participants referenced different components of their partner models of agents changing through the programming activity (relative to the total number of partner model related tags). Competence and reliability were referenced most.

Trust

As with overall participants' partner models, there were few significant differences in the quantitative data about participants' trust of agents between the pre-, mid- and post-surveys. (There were many differences for subsets of the participants, however, as described in following sections.) For participants overall (and all major subsets too, including those from WEIRD and non-WEIRD countries, children, and parents) prior to, during and after the workshops, Google, Alexa and newspapers were generally trusted significantly more than both parents and friends to report correct information. Figure 3-13 shows this trend for participants overall. The differences between systems (i.e., Google, Alexa, the newspaper, parents and children) were typically significant for all major subsets too, with a few exceptions (e.g., for children after the workshops, there was not a significant difference between trust of newspapers and parents). Many of the differences and corresponding significance levels for the major subsets are shown in Appendix D. In all cases, the means for trust of Google, Alexa and newspapers were higher than that of parents and friends. Additionally, the means for trust of parents was always greater than that of friends, although this difference was not always significant (see the post-survey in Figure D-2). In other words, people tended to trust technology more than people, and their parents more than friends for correct information.

When asked in a long-answer question why they trusted or distrusted agents to provide correct information, participants generally provided answers for why they distrusted

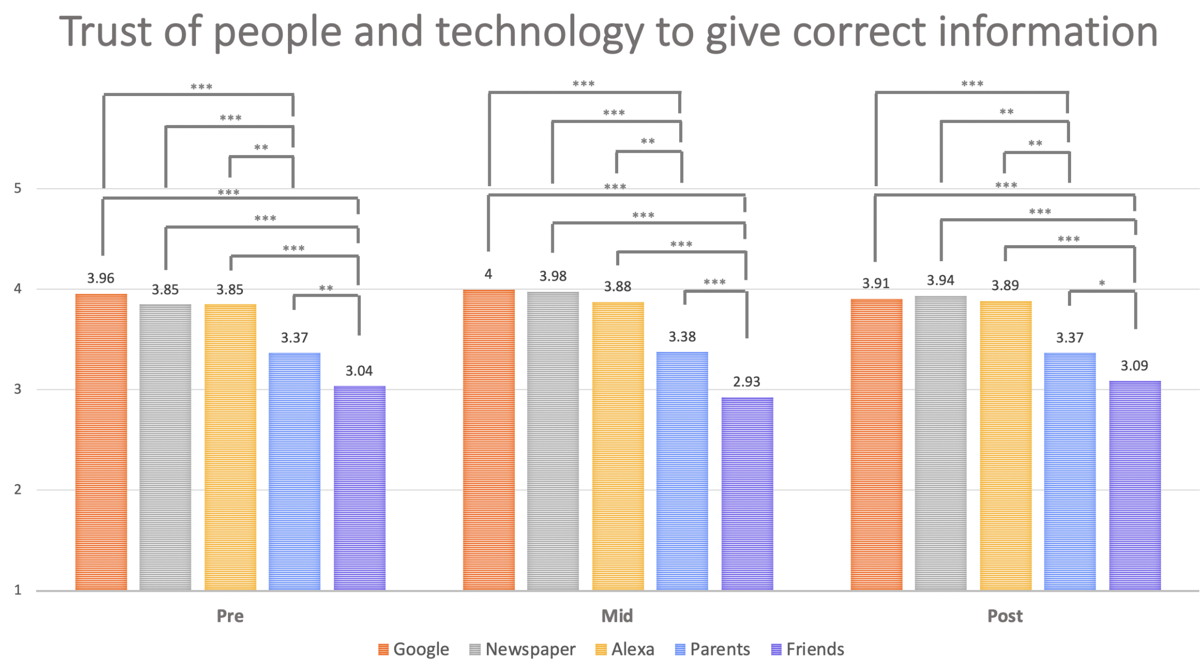


Figure 3-13: The mean responses for participants overall when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. Participants tended to trust technology more than people throughout the workshops.

agents. Table 3.6 shows the results from before the programming activity, and Table 3.7 show the results from after the programming activity. Both tables show how a large majority of answers from each of the subsets indicated distrust. After the programming activity, the percent of answers indicating distrust decreased slightly for all subsets.

Figure 3-14 compares participants' reasoning for their opinions on trust before and after the programming activity. Both before and after the activity, participants most often mentioned the source of the agent's information, including human data, the internet and other sources, as reasons for their opinions on trust. Table 3.19 describes the various themes we developed from participants' answers. After the programming activity, there were fewer ambiguous reasons and more reasoning with respect to how agents are programmed.

By categorizing participants' reasons for trust into the four characteristics described by McKnight and Chervany's model (competence, benevolence, integrity and predictability [107]), we found participants' reasoning leaned towards the aspect of competence for nearly all subsets on both the pre- and mid-survey. Table 3.8 and 3.9 show this, as well as how we found no responses indicating participants considered the benevolence aspect of trust with respect to conversational agents.

Table 3.6: Percent of long-answer responses indicating participants generally trusted, distrusted, or were unsure about whether they trusted conversational agents on the pre-survey. The cells in bold indicate the largest percentages for different subsets.

Subset	Trustworthy	Not trustworthy	Unsure
Overall participants	19%	**72%**	9%
Non-WEIRD	15%	**70%**	15%
WEIRD	22%	**74%**	4%
Children	20%	**72%**	8%
Parents	17%	**72%**	11%

Table 3.7: Percent of long-answer responses indicating participants generally trusted, distrusted, or were unsure about whether they trusted conversational agents on the mid-survey. The cells in bold indicate the largest percentages for different subsets.

Subset	Trustworthy	Not trustworthy	Unsure
Overall participants	22%	**64%**	14%
Non-WEIRD	13%	**67%**	20%
WEIRD	29%	**62%**	10%
Children	27%	**59%**	14%
Parents	14%	**71%**	14%

Table 3.8: Percent of long-answer responses indicating different aspects of McKnight and Chervany's trust model when participants discussed their opinions on trust of conversational agents on the pre-survey. The cells in bold indicate the largest percentages for different subsets.

Subset	Competence	Integrity	Predictability	Benevolence
Overall	**39%**	25%	36%	0%
Non-WEIRD	**36%**	29%	**36%**	0%
WEIRD	**41%**	23%	36%	0%
Child	34%	30%	**36%**	0%
Parent	**48%**	17%	35%	0%

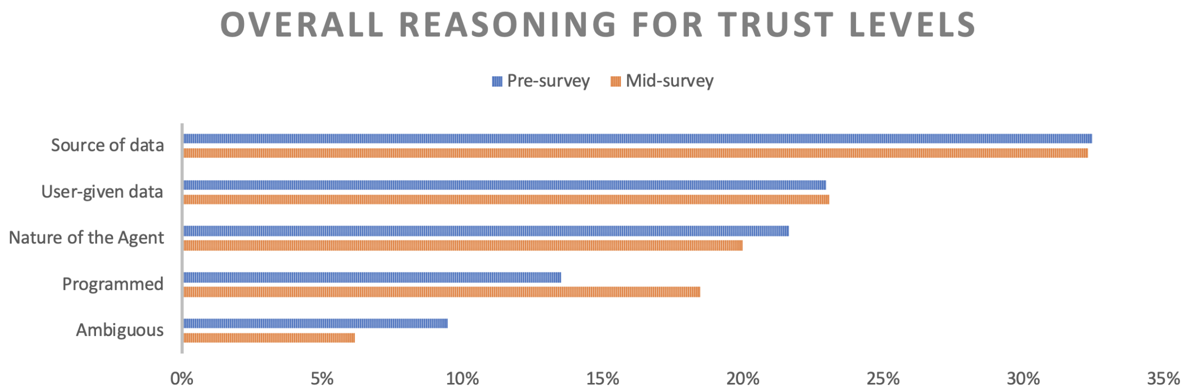


Figure 3-14: Overall participants' responses to the question asking about their reasoning for their opinions on trust of agents in terms of counted tag frequency. Table 3.19 provides descriptions of each concept. Participants most often mentioned the source of where agents obtain their data when describing their trust of agents.

Table 3.9: Percent of long-answer responses indicating different aspects of McKnight and Chervany's trust model when participants discussed their opinions on trust of conversational agents on the mid-survey. The cells in bold indicate the largest percentages for different subsets.

Subset	Competence	Integrity	Predictability	Benevolence
Overall	**43%**	23%	34%	0%
Non-WEIRD	37%	22%	**41%**	0%
WEIRD	**47%**	24%	29%	0%
Child	**37%**	26%	**37%**	0%
Parent	**52%**	17%	30%	0%

About a quarter of participants' long-answer responses indicated they felt their opinions on the trustworthiness of agents changed through the programming activity. Table 3.10 shows the results for the major subsets. As shown in Figure 3-15 participants most often cited the source of the agents' information, including human data, the internet and other sources, as being the reason for their opinion changing. Participants' responses in this category (as well as the "Programmed" category) often alluded to how they had learned about the agents' source of data or programming. The next most often cited reason was participants' personal (or general) learning (e.g., "after learning, it does help me realize the feature and the potential that these 'agents' have.").

When discussing changes in trust of conversational agents, participants cited predictability most often with respect to McKnight and Chervany's trust model. As with

Table 3.10: Percent of long-answer responses indicating participants felt their opinions on the trustworthiness of agents changed through the programming activity. The cells in bold indicate the largest percentages for different subsets.

Subset	Changed	Did not change	Ambiguous
Overall participants	24%	**63%**	13%
Non-WEIRD	32%	**53%**	16%
WEIRD	16%	**74%**	11%
Children	23%	**68%**	9%
Parents	25%	**56%**	19%

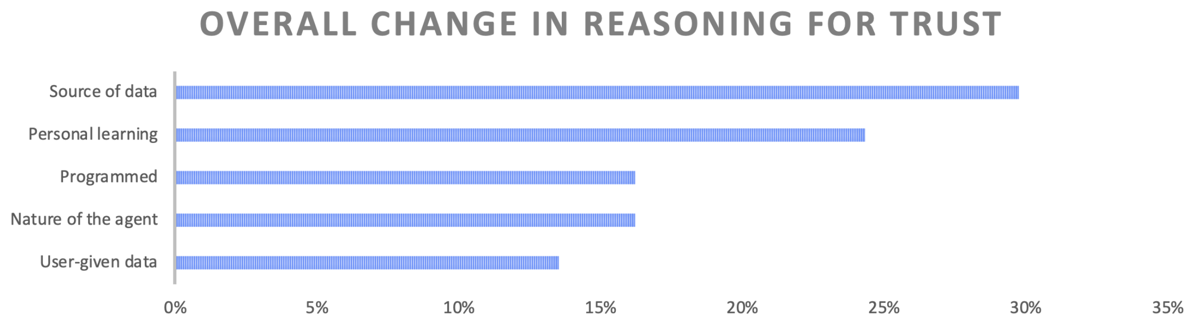


Figure 3-15: Participants' responses to the question asking about their reasoning for any change in opinion on trust of agents in terms of percent tag frequency. Table 3.19 provides descriptions of each concept. Participants most often referenced learning about the source of agents' data (e.g., the internet, human data, etc.) as being a reason for changes in their trust.

discussing their general opinions of trust, we did not find participants discussed benevolence when discussing changes in trust. Table 3.11 shows these results.

Challenging Concepts

We also asked participants which concepts they found most difficult to learn after the programming activity (as shown on the mid-survey in Appendix B). Table 3.12 shows the concepts participants could choose from and the corresponding descriptions on the survey question. Overall, the top three concepts participants found most difficult were: (1) Training, (2) Terminology and (3) Turn-taking. Figure 3-16 shows all of the concepts participants chose as most difficult in order from most to least frequent.

In terms of other comments from the participants on what was challenging, participants mentioned debugging (39%), AI-related challenges (27%), other coding challenges

Table 3.11: Percent of long-answer responses indicating different aspects of McKnight and Chervany's trust model when participants discussed changes in their trust of conversational agents through the programming activity. The cells in bold indicate the largest percentages for different subsets.

Subset	Competence	Integrity	Predictability	Benevolence
Overall	32%	29%	**39%**	0%
Non-WEIRD	**40%**	20%	**40%**	0%
WEIRD	28%	33%	**39%**	0%
Child	25%	0%	**75%**	0%
Parent	**54%**	8%	38%	0%

Table 3.12: Descriptions of concepts from the workshops participants found most challenging. These descriptions were used in the mid-survey.

Concept	**Description**
Training	How agents need to be trained with examples or "utterances"
Intent	How you can say different things (an "intent") to an agent and it still understands
Agent Modularization	How you need to say the name of the Alexa skill ("invocation name") for it to understand
Entities	How to make Alexa get (or use) specific information, like a number, from the user (i.e., using "slots", which are also known as entities)
Events	How to make Alexa do or say things (making the "endpoint function")
Terminology	The terminology (e.g., what "invocation name" or "intent" means)
Testing	How to type/say something so that Alexa understands ("testing")
Turn-taking	How to get Alexa to ask a follow-up question (the "ask" block)
Other	(Participants can enter in their own description of challenges)

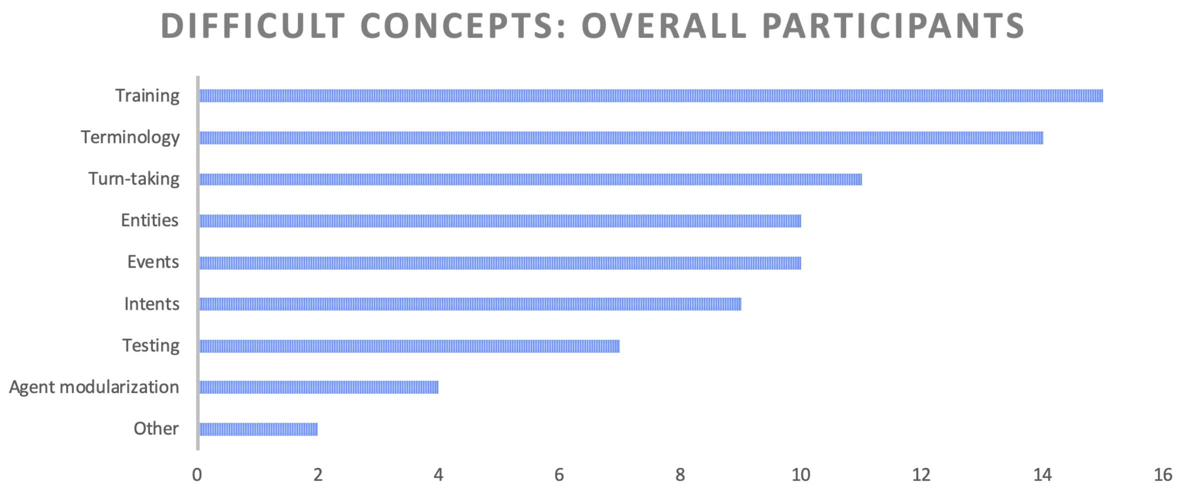


Figure 3-16: Participants' responses overall to the question asking which concepts were most difficult to learn in terms of count frequency. Table 3.12 provides descriptions of each concept. Participants most often referenced Training as being the most challenging concept.

(15%), and additional uncategorized challenges (19%).

Ideal Agents

In terms of qualitative results, participants described their ideal conversational agents in the Miro activity (as described in Section 3.2.1) with more task-oriented (75%) than non-task oriented (or socially-oriented; 25%) language, as shown in Figure 3-17. Example task and non-task oriented phrases from participants' responses can be found in Table 3.13. All subsets of participants analysed (e.g., children, parents, those from WEIRD countries, etc.) also showed this task-orientation, albeit with different proportions. Participants also used slightly more human-like (55%) than artificial (45%) descriptions (as shown in Figure 3-18) for their agents. However, this was not the case for all subsets of participants, including how children from non-WEIRD countries described their ideal agents with more artificial descriptions (59%). Example human-like and artificial-related phrases can be found in Table 3.14.

Overall, the frequency of tags describing ideal agents from most to least frequent were: *basic features*, *approachable/friendly*, *personalized*, *proactive*, *fun* tied with *addresses concerns*, *convenient*, *culturally intelligent*, and *familiar or pop-culture related*, as shown in Figure 3-19. The descriptions of these tags can be found in in Table 3.15. The frequency order for the major subsets of participants (i.e., children, parents, participants from non-

Table 3.13: Descriptions of the themes, non-task oriented and task-oriented identified in participants' responses to questions about their ideal conversational agents.

Tag	Definition	Example utterances
Non-task oriented	Phrases and features that are relational or socially-oriented, and don't attempt to achieve a specific goal	"Welcome back.", "How is your day?", "Have a conversation", "Hey! What is your plan for today?", "How are you feeling today"
Task-oriented	Phrases and features that have to do with completing a specific goal	"Connect and bring interactivity to a person's smart home (i.e. lights, tvs, fridges, speakers, etc.)", "Weather local updates", "How can I assist you today?", "Check your wellbeings", "Current events"

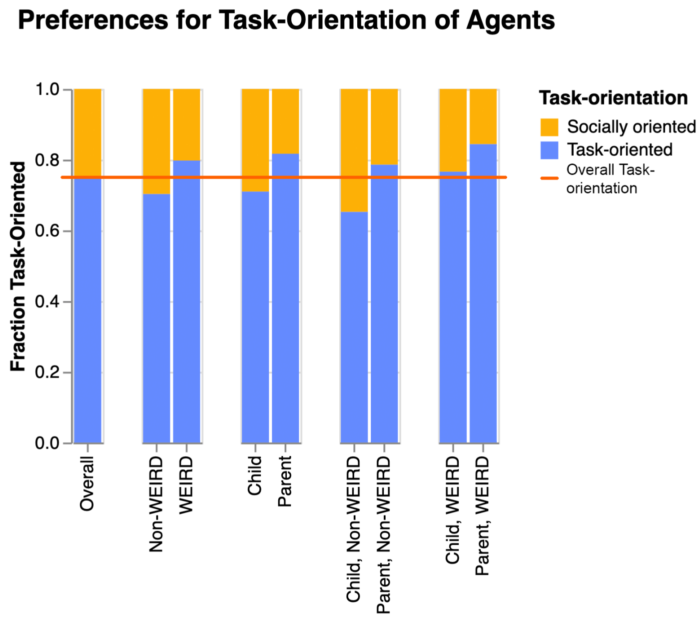


Figure 3-17: The number of phrases indicating a preference for either task-oriented or non-task oriented (i.e., socially-oriented) agents normalized and grouped by various subsets of the participants. The data was from participants' descriptions of their "ideal" agents. The orange line indicates the task-orientation from all participants' comments. All subsets had a preference for task-oriented agents; however, those from non-WEIRD countries and children had relatively higher preferences for socially oriented agents.

WEIRD countries and participants from WEIRD countries) was similar to this overall order. Any major differences are described in the respective sections.

Self-Efficacy and Identity

There were many significant differences in overall participants' self-efficacy and identities as programmers throughout the workshops. For example, participants' confidence in terms of feeling like a programmer significantly increased after the programming ac-

Table 3.14: Descriptions of the themes, human-like and artificial, which were identified in participants' responses to questions about their ideal conversational agents.

Tag	Definition	Example utterances
Human-like	Looks or sounds human, or has human-like emotions	"I think they should have a better voice with more emotion on it", "expressive", "not feels like robots", "The voice with emotion"
Artificial	Looks or sounds un-human, artificial or robotic	"Like a robot, but not human like otherwise it would be a bit creepy.", "not human like", "Alien", "Probably just a program."

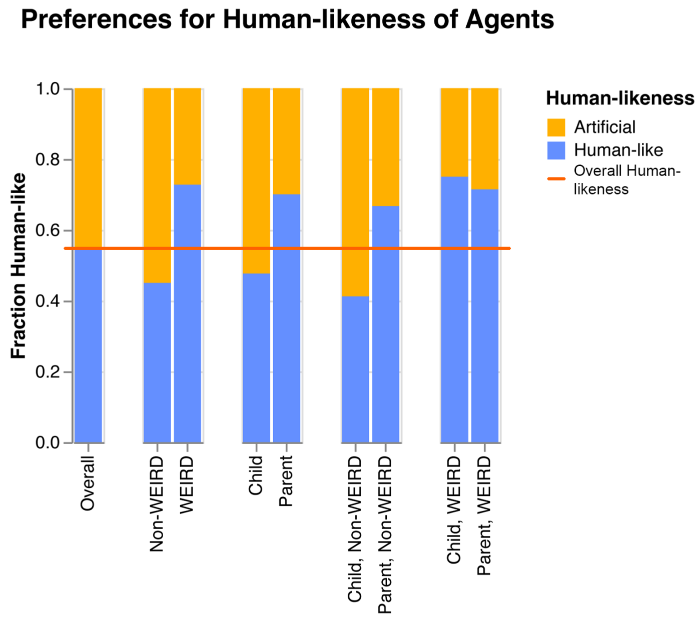


Figure 3-18: The number of phrases indicating a preference for either human-like or artificial (e.g., robotic) agents normalized and grouped by various subsets of the participants. The data was from participants' descriptions of their "ideal" agents. The orange line indicates the human-likeness preference from all participants' comments. Generally, those from non-WEIRD countries and children had a slight preference for artificial (over human-like) agents. Note that there were few comments about human-likeness or artificiality from parents from non-WEIRD countries, and from children and parents from WEIRD countries (as alluded to in Figure 3-25).

tivity, and remained significantly high after the societal impact activity too (pre/mid: $\bar{x}$=2.79,3.21; t(38)=-4.02; p=2.66×10^{-4}; pre/post: $\bar{x}$=2.79,3.39; t(32)=-4.94; p=2.35×10^{-5}). There was a similar result for participants' confidence in creating their own technology projects (pre/mid: $\bar{x}$=3.21,3.62; t(38)=-3.58; p=.001; pre/post: $\bar{x}$=3.18,3.76; t(32)=-3.98; p=3.67×10^{-4}). Participants' confidence in terms of being able to make an impact in their community or the world using technology only significantly increased after the societal impact activity (mid/post: $\bar{x}$=3.64,3.94; W(32)=18; p=.039).

Table 3.15: Descriptions of the main themes identified in participants' responses to questions about their ideal conversational agents.

Tag	Definition	Example utterances
Convenient	Portable, ambient, can be found many places, or can do things at a distance, like sending a car to get gas	"Something easy to bring around so it can easily be relocated to wherever the user needs it", "Some sort of an accessory (most likely glasses)", "A vast network of smart speakers and displays, similar to current times"
Personalized	Able to be customized by the user or automatically personalized to the user	"Impersonate my voice, and get on the phone with people I know.", "It would sing happy Birthday with the persons name.", "insert name"
Proactive	Will *initiate* conversation	"Hey! What is your plan for today?", "How can I assist you today?", "How is your day?", "How are you feeling today"
Approachable/ Friendly	Has friendly, warm, approachable, helpful features, phrases, or appearances	"makes the user feel comfortable talking to the bot.", "maybe positive uplifting notes like 'you can do it!!!' ", "Friendly reminders", "cute robot"
Familiar or pop-culture related	References to pop-culture or familiar concepts	"BB8", "Microsoft paper clip", "R2D2", "Stevie Nicks"
Fun	Amusing, humorous, enjoyable, etc.	"a fancy penguin with a bowtie and a monocle", "Toonish 3D image on a screen", "Video game partner", "jokes", "It would sing happy Birthday"
Culturally intelligent	Can adapt to different cultures/languages/accents, or knows how to be respectful/ considerate	"say local words without an accent", "Add persian language", "Polite responses", "Many more languages than main languages"
Addresses concerns	Features/phrases/ideas that address specific concerns, like security, mental health and physical health	"encourage the listener be their best self and be emotionally and mentally stable", "Don't give it too much intelligence otherwise it'll summon something unholy", "covid-19 updates"
Basic features	Common functions or appearances that current conversational agents already have or look like	"updates on news", "fun facts", "Stock market analysis", "reminders", "A vast network of smart speakers and displays, similar to current times"

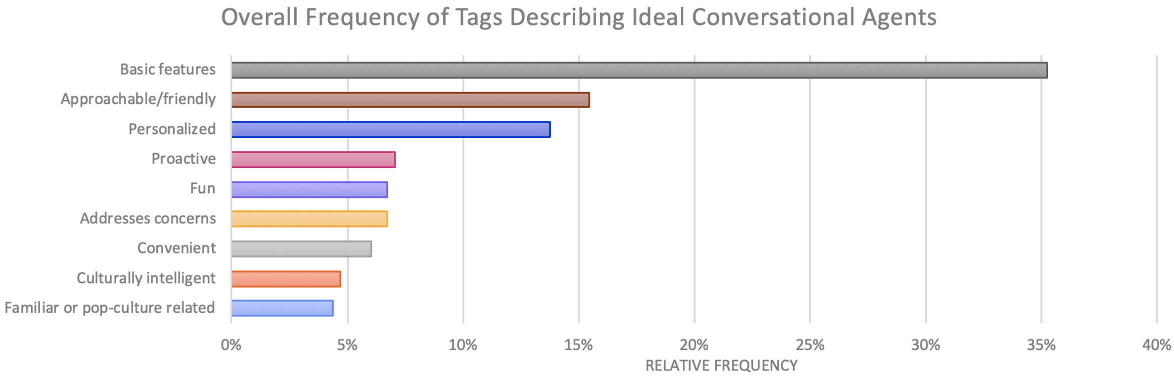


Figure 3-19: Bar chart showing the frequency of phrases coded with particular tags. The data tagged was from overall participants' descriptions of their "ideal" agents. To obtain the relative tag frequency, the raw count of each tag was normalized over the total number of tags. Table 3.15 shows the definitions of each tag. Overall, participants mentioned agents having basic features, like being able to set reminders or provide news updates, most often. They also frequently mentioned agents having a friendly personality and being personalized.

3.4.2 Participants from WEIRD vs. Non-WEIRD Countries

Demographics and Tutorial Completion

Twenty-one participants from non-WEIRD countries and 25 from WEIRD countries filled out the pre-survey. There was no significant difference in the number of tutorials participants from non-WEIRD vs. WEIRD countries completed.

Notable Similarities. There was no significant difference between the amount of prior programming experience those from WEIRD vs. non-WEIRD countries had, or the number of tutorials they completed by the end of the workshops. There was also no significant difference in whether or not they had prior experience with a conversational agent or prior experience learning about AI.

Notable Differences. Participants from non-WEIRD countries had experienced fewer types of conversational agents than those from WEIRD countries ($\bar{x}$=1.05,2.20; U(44)=-4; p=7.52×10^{-5}) and specifically also had less prior experience with Amazon Alexa ($\bar{x}$=0.64,0.00; U(44); p=3.79×10^{-6}). (This is despite the fact that there was no significant difference between whether participants had experienced *any type* of conversational agent.)

Unsurprisingly, participants from non-WEIRD countries spoke English less as their first language than those from WEIRD countries ($\bar{x}$=0.33,0.84; U(44)=129.5; p=2.74 × 10^{-4}), and spoke to conversational agents in their first language less ($\bar{x}$=0.71,0.92; U(44)=208.5; p=.036).

Parents from non-WEIRD countries had less experience learning about AI than parents from WEIRD countries ($\bar{x}$=0.13,0.63; U(19)=26.5; p=.017). They were also younger than parents from WEIRD countries ($\bar{x}$=44.25,48.25; U(17)=21; p=.031).

Partner Model

Participants from WEIRD countries thought Alexa was less competent after the programming activity than before (pre/mid: $\bar{x}$=2.43,2.95; t(20)=-2.33; p=.030). This resulted in a significant difference between how participants from WEIRD and non-WEIRD countries felt about Alexa's competence after the programming activity ($\bar{x}$=3.00,2.11; U(38)=106.5; p=.004), as shown in Figure 3-20. After the entire workshops, children from non-WEIRD countries thought Alexa seemed more competent than before (pre/post: $\bar{x}$=3.20,2.30; W(9)=0; p=.0067).

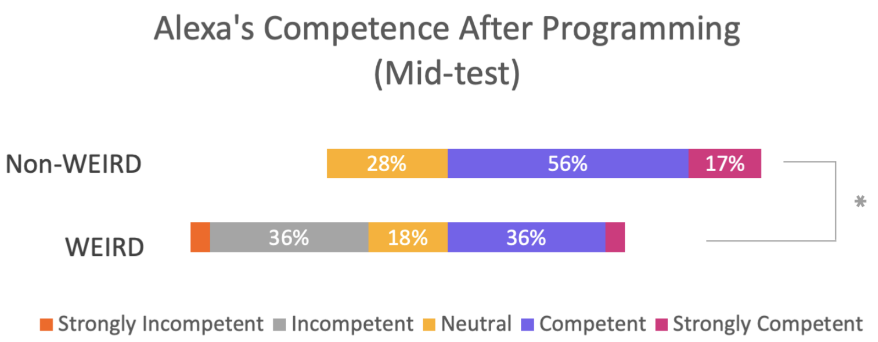


Figure 3-20: Distribution of responses from participants from non-WEIRD and WEIRD countries when asked to rate Alexa's competence. These results are from 5-point Likert scale question given after the programming activity.

After the programming activity, participants from WEIRD countries also thought Alexa was more of a peer than an authority figure than what participants from non-WEIRD thought ($\bar{x}$=3.64,3.00; U(38)=134; p=.036), as shown in Figure 3-21. At this point, those from WEIRD countries also thought Alexa was less dependable than what those from non-WEIRD countries thought ($\bar{x}$=3.23,3.78; U(38)=122.5; p=.014), as shown in Figure 3-22. This may have been due to how participants from non-WEIRD countries' perspectives shifted through the programming activity, as they thought Alexa seemed *more* dependable afterwards than before (pre/mid: $\bar{x}$=3.39,3.78; W(17)=0; p=.020). Participants from non-WEIRD countries' perspectives also shifted in terms of flexibility, as they thought Alexa seemed more flexible afterwards than before (pre/mid: $\bar{x}$=2.89,3.33; W(17)=4.5; p=.021).

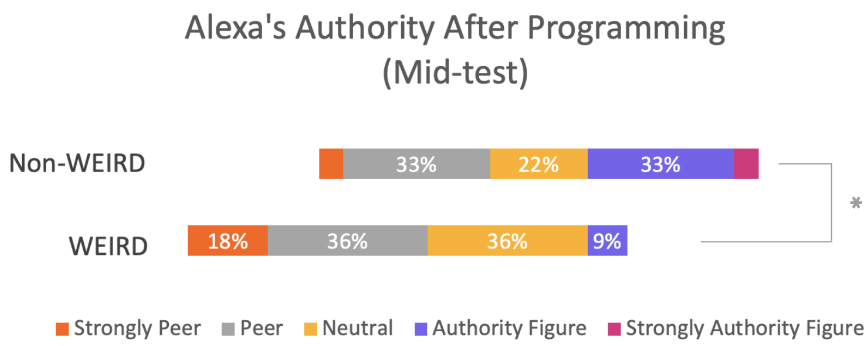


Figure 3-21: Distribution of responses from participants from non-WEIRD and WEIRD countries when asked to rate Alexa's authority. These results are from 5-point Likert scale question given after the programming activity.

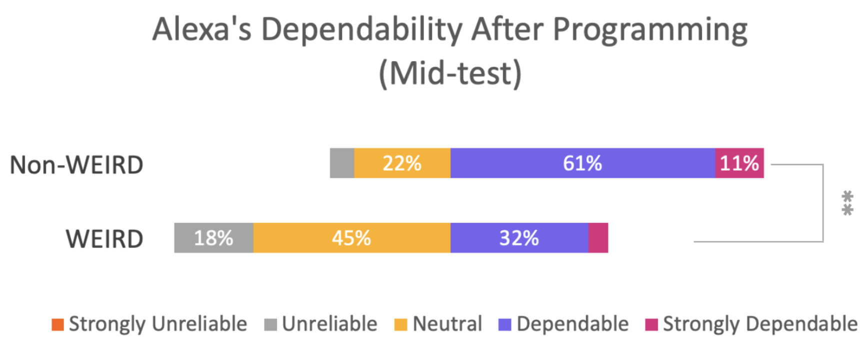


Figure 3-22: Distribution of responses from participants from non-WEIRD and WEIRD countries when asked to rate Alexa's dependability. These results are from 5-point Likert scale question given after the programming activity.

More specifically, children from WEIRD countries thought Alexa (or agents they had interacted with) seemed more flexible than children from non-WEIRD countries did prior to the workshops ($\bar{x}$=3.50,2.77; U(25)=55; p=.037).

Figure 3-23 shows different aspects of partner models, which participants from non-WEIRD and WEIRD countries referenced when describing how they felt their perceptions of agents changed through the workshops. Participants from non-WEIRD countries referenced agents' dependability and interactivity more than those from WEIRD countries did; whereas participants from WEIRD countries referenced agents' flexibility and human-likeness more than those from non-WEIRD countries did.

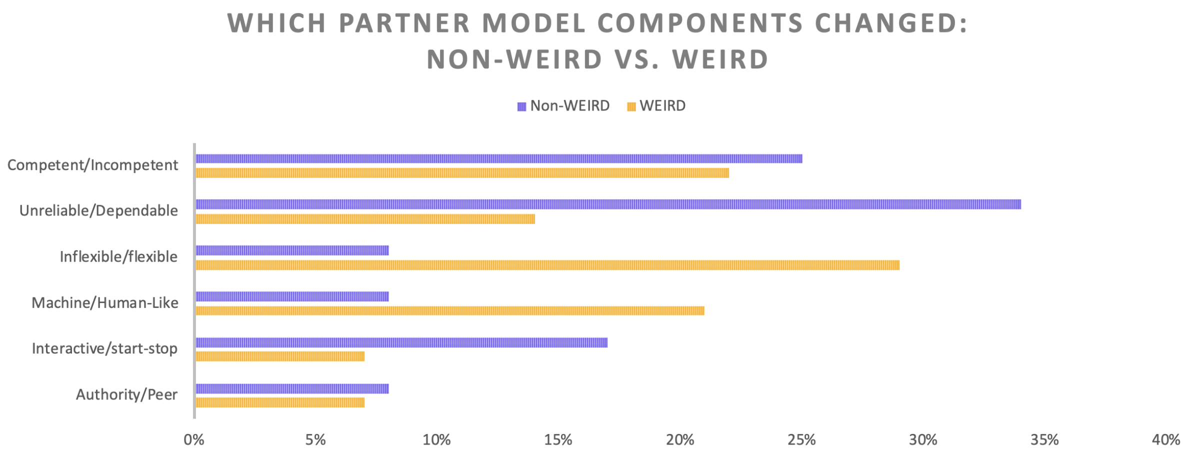


Figure 3-23: The frequency participants from non-WEIRD and WEIRD countries referenced different components of their partner models of agents changing through the programming activity (relative to the number of partner model related tags per subset).

Self-Efficacy and Identity

There were no significant differences found in terms of identifying as a programmer and participants' self-efficacy from WEIRD vs. non-WEIRD countries overall. However, differences were found for *just children*. Although prior to the workshops, there was no significant difference between whether children from WEIRD vs. non-WEIRD countries felt confident they could make an impact in their community or the world using technology, after the programming activity ($\bar{x}$=4.08,3.45; U(22)=43; p=.045) and the entire workshops ($\bar{x}$=4.45,3.50; U(19)=22; p=.0078), children from WEIRD countries felt more confident in this respect.

Trust

Participants from WEIRD countries' feelings of trust changed through the workshops. They trusted Alexa (pre/post: $\bar{x}$=4.00,3.75; W(15)=0; p=.046), their parents (pre/post: $\bar{x}$=3.50,3.25; W(15)=0; p=.046) and their friends (pre/post: $\bar{x}$=3.19,2.90; W(20)=0; p=.034) less to give correct information after the workshops than before. This was unlike the feelings of those from non-WEIRD countries, where there were no significant differences between pre- and post-survey trust results. Figure 3-24 shows participants from non-WEIRD and WEIRD countries' reasoning for their change in trust. Those from WEIRD countries most often mentioned where agents obtained their data from (e.g., from the internet) when describing any change in trust; whereas those from non-WEIRD countries most often mentioned generally learning something from the workshops.

After the programming activity, participants from WEIRD countries thought agents would report correct information less than participants from non-WEIRD countries ($\bar{x}$=2.82,2.33; U(38)=123; p=.011). More specifically, children from non-WEIRD countries thought agents were *more* trustworthy after the programming activity (pre/mid: $\bar{x}$=3.27,3.64; W(10)=0; p=.046).

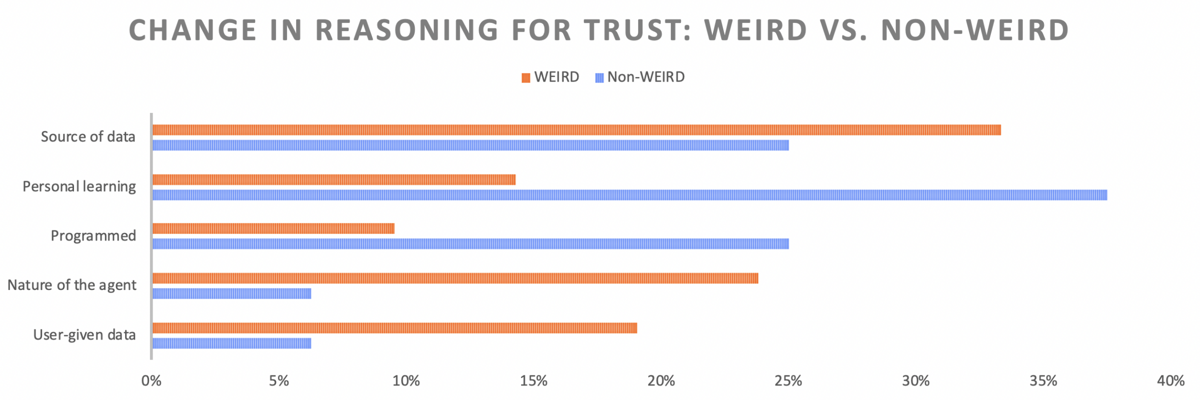


Figure 3-24: Participants from non-WEIRD and WEIRD countries' responses to the question asking about their reasoning for any change in opinion on trust of agents in terms of percent tag frequency. Table 3.19 provides descriptions of each concept.

Another difference includes how in the pre-survey, children from WEIRD countries trusted newspapers to report correct information more than children from non-WEIRD countries ($\bar{x}$=4.29,3.62; U(25)=54; p=.022).

Ideal Agents

As shown in Figure 3-17, when commenting on their ideal conversational agents, participants from WEIRD countries had relatively more task-orientation (80%) than participants from non-WEIRD countries (70%). As shown in Figure 3-25, participants from non-WEIRD countries commented more often (65%) than those from WEIRD countries (35%) on artificiality and human-likeness of conversational agents. Participants from non-WEIRD countries commented relatively more on how conversational agents should be *artificial* (55%) than those from WEIRD countries (27%) (as shown in Figure 3-18).

As shown in Figure 3-26, people from WEIRD countries tended to focus more on *basic (already common conversational agent) features* and *pop-culture or familiar features* than those from non-WEIRD countries, whereas people from non-WEIRD countries tended to

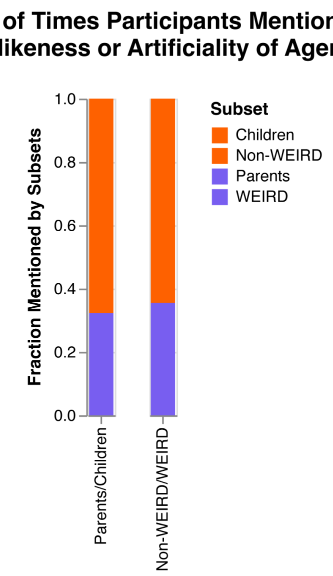


Figure 3-25: The number of comments on agent artificiality and human-likeness normalized and grouped by subsets of the participants. Children and those from non-WEIRD countries tended to comment on artificiality and human-likeness more than their counterparts.

focus more on *proactivity*, *cultural intelligence* and *addressing concerns* than those from WEIRD countries.

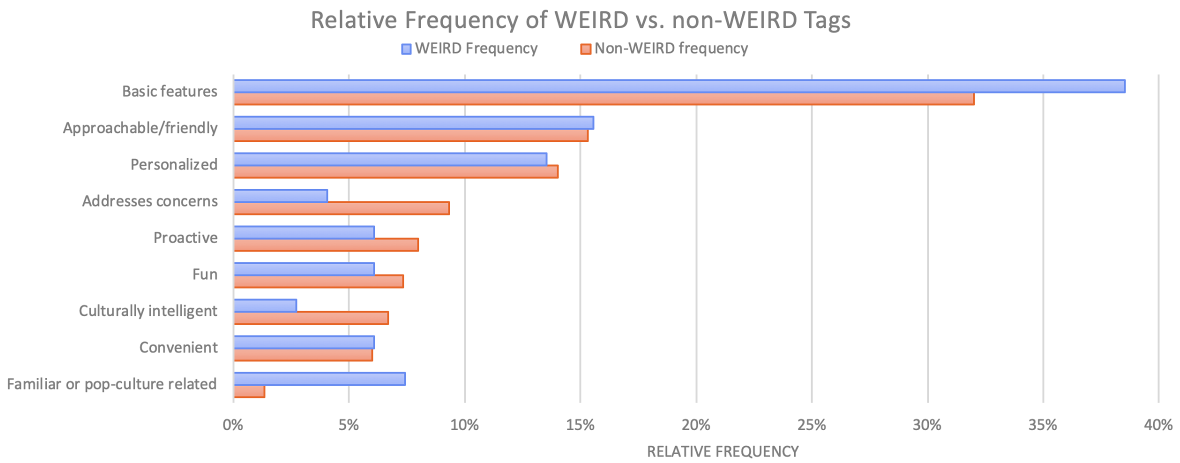


Figure 3-26: Bar charts comparing the frequency of phrases coded with particular tags from responses from those from non-WEIRD and WEIRD countries. The data tagged was from participants' descriptions of their "ideal" agents. To obtain the relative tag frequency, the raw count of each tag was normalized over the total number of tags from that particular subset (e.g., the number of *Fun* tags from data from those from non-WEIRD countries was normalized over the total number of tags from data from those from non-WEIRD countries). Table 3.15 shows the definitions of each tag.

This figure compared to Figure 3-19 also shows how the order of tags from highest to lowest frequency for participants from WEIRD and non-WEIRD countries was similar to the overall frequency order. The main differences in the non-WEIRD frequency order included: (1) the tag order of *addresses concerns* and *proactive* being swapped, with *addresses concerns* being slightly more common than *proactive*, and (2) the tag order of *culturally intelligent* and *convenient* being swapped, with *culturally intelligent* being slightly more common than *convenient*. The main differences in the WEIRD frequency order included: (1) *familiar or pop-culture related* shifting to fourth most frequent (from last), (2) *proactive* shifting one place less frequent, (3) *addresses concerns* shifting two places less frequent, and (4) *culturally intelligent* shifting one place less frequent.

Challenging Concepts

There were differences in terms of which concepts participants from non-WEIRD and WEIRD countries felt were most difficult to learn. Figure 3-27 shows these differences. For example, those from non-WEIRD countries more often cited Training and Events as difficult concepts than those from WEIRD countries; whereas those from WEIRD countries more often cited Testing and Turn-taking, and more often described other concepts. Otherwise, the relative frequencies were quite similar.

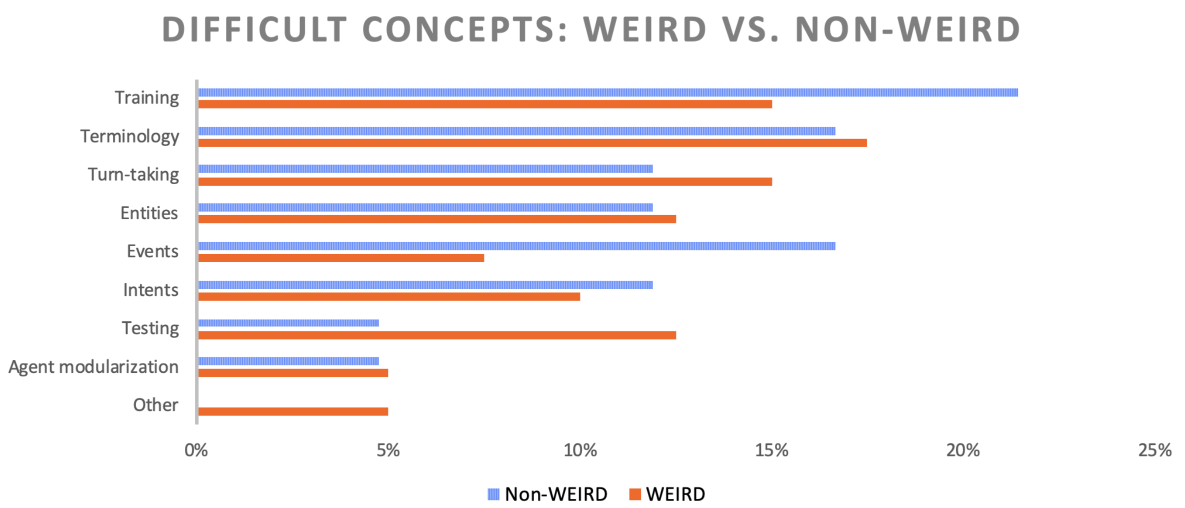


Figure 3-27: Participants from WEIRD vs. non-WEIRD countries' responses to the question asking which concepts were most difficult to learn. The graph shows this in terms of relative frequency. Table 3.12 provides descriptions of each concept.

3.4.3 Parents vs. Children

Demographics and Tutorial Completion

Of those who completed the pre-survey, there were 19 parents (including one who was also a grandparent) and 27 children.

Notable Similarities. There was no significant difference between the amount of prior experience learning about AI for children and parents from WEIRD countries, although children from non-WEIRD countries had more prior experience learning about AI than parents from non-WEIRD countries (as described below).

Notable Differences. Children had more prior experience programming than parents ($\bar{x}$=1.59,0.79; U(44)=144.5; p=.0026). Children from non-WEIRD countries had more prior experience learning about AI ($\bar{x}$=0.62,0.13; U(19)=27; p=.017) than parents from non-WEIRD countries (although this was not so for those from WEIRD countries). Children from WEIRD countries completed significantly more tutorials than parents from WEIRD countries ($\bar{x}$=2.14,1.14; U(16)=18.5; p=.0.025).

Partner Model

Before ($\bar{x}$=2.74,2.11; U(44)=167; p=.018) and after ($\bar{x}$=2.79,2.13; U(38)=112; p=.0093) the programming activity, children thought Alexa was more human-like than parents did. They also thought Alexa was warmer than their parents did before ($\bar{x}$=2.70,3.37; U(44)=170.5; p=.021), during ($\bar{x}$=2.96,3.56; U(38)=129.5; p=.034) and after ($\bar{x}$=2.62,3.50; U(33)=81.5; p=.011) the workshops. After the programming activity, they thought Alexa was more dependable than their parents did ($\bar{x}$=3.82,3.14; U(16)=21; p=.039).

Figure 3-28 shows different aspects of partner models, which children and parents referenced when describing how they felt their perceptions of agents changed through the workshops. Children referenced agents' competence, dependability and flexibility more than parents did; whereas parents referenced agents' human-likeness, interactivity and authority more than children did.

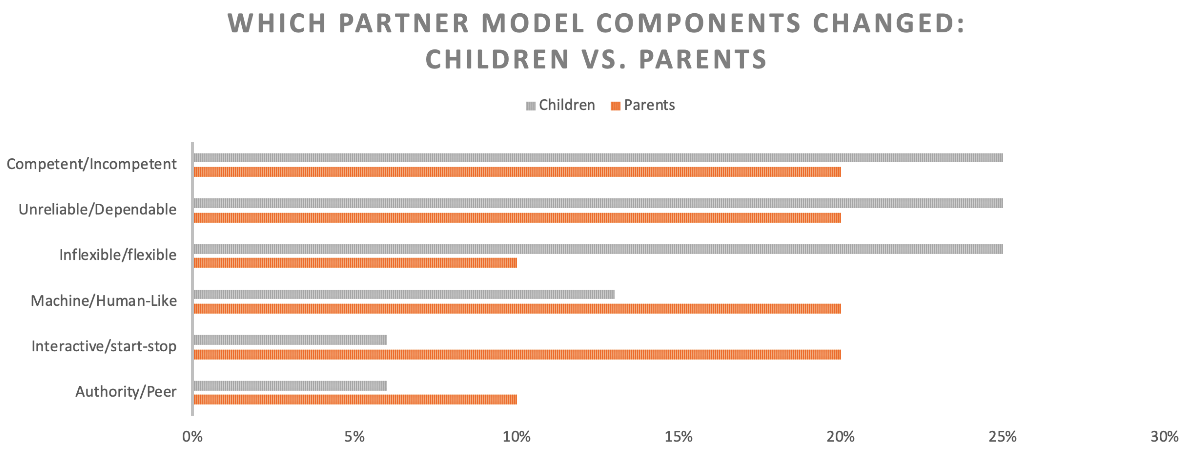


Figure 3-28: The frequency children and parents referenced different components of their partner models of agents changing through the programming activity (relative to the number of partner model related tags per subset).

Self-Efficacy and Identity

Children saw themselves more as programmers than parents prior to the workshops ($\bar{x}$=3.56,1.89; U(44)=88; p=5.98 $\times$ 10^{-5}), during ($\bar{x}$=3.83,2.31; U(38)=71.5; p=3.32 $\times$ 10^{-4}), and after ($\bar{x}$=4.00,2.43; U(33)=38.5; p=8.35 $\times$ 10^{-5}). This aligns with how children had more prior programming experience, as mentioned in Section 3.4.3. They also were more confident they could design and create their own technology project than parents prior to the workshops ($\bar{x}$=3.59,2.63; U(44)=145.5; p=.0057), during ($\bar{x}$=3.96,3.19; U(38)=126.5; p=.030), and after ($\bar{x}$=4.05,3.21; U(33)=85.5; p=.014). After the workshops, children from WEIRD countries were more confident they could make an impact in their community or the world using technology than parents from WEIRD countries ($\bar{x}$=4.45,3.57; U(16)=16; p=.017). This was not the case for children vs. parents from non-WEIRD countries.

Children felt more confident they could make an impact in their community or in the world using technology after the the entire workshop than before (pre/post: $\bar{x}$=3.50,4.00; W(19)=0; p=.014). This was not the case for parents; although it was the case for participants overall, as mentioned in Section 3.4.1.

Trust

As shown in Figure 3-29, after the programming activity, children thought Alexa was more correct than parents did ($\bar{x}$=4.04,3.63; U(38)=127.5; p=.023). Children also thought agents would report correct information more after the societal impact activity than before (mid/post: $\bar{x}$=2.60,2.35; t(19)=2.52; p=.021).

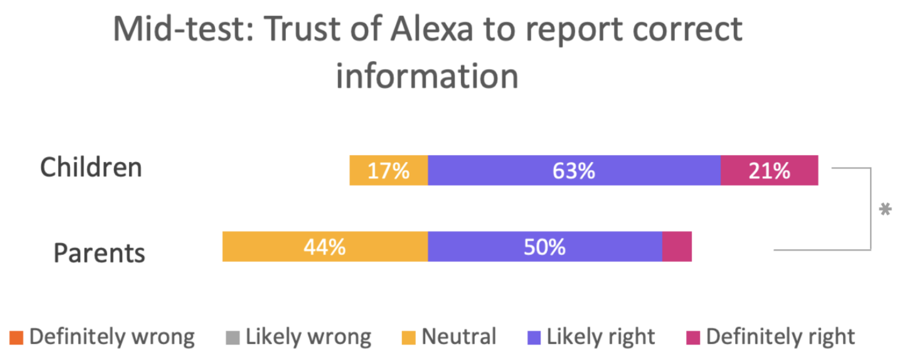


Figure 3-29: Distribution of responses from children and parents when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given after the programming activity.

Figure 3-30 shows children and parents' reasoning for their change in trust. The reasoning frequencies are similar; however, children mentioned the general nature of the agent more than parents did, and parents mentioned how agents are programmed more than children did. Table 3.19 describes participants' reasoning in more detail.

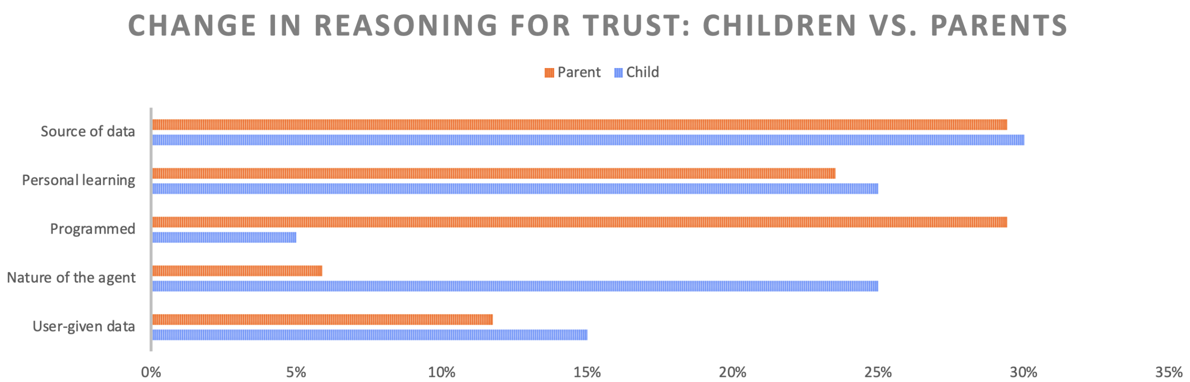


Figure 3-30: Children and parents' responses to the question asking about their reasoning for any change in opinion on trust of agents in terms of percent tag frequency. Table 3.19 provides descriptions of each concept.

Ideal Agents

As shown in Figure 3-17, when commenting on their ideal conversational agents, parents had relatively more task-orientation (82%) than children (71%). As shown in Figure 3-25, children commented more (68%) than parents (32%) about artificiality and human-likeness of conversational agents. Children commented relatively more on how conversational agents should be *artificial* (52%) than parents (30%) (as shown in Figure 3-18).

As shown in Figure 3-31, parents tended to focus more on *personalized features* and *pop-culture or familiar features* than children, whereas children tended to focus more on *fun features*, *approachable/friendly features*, and *addressing concerns* than parents.

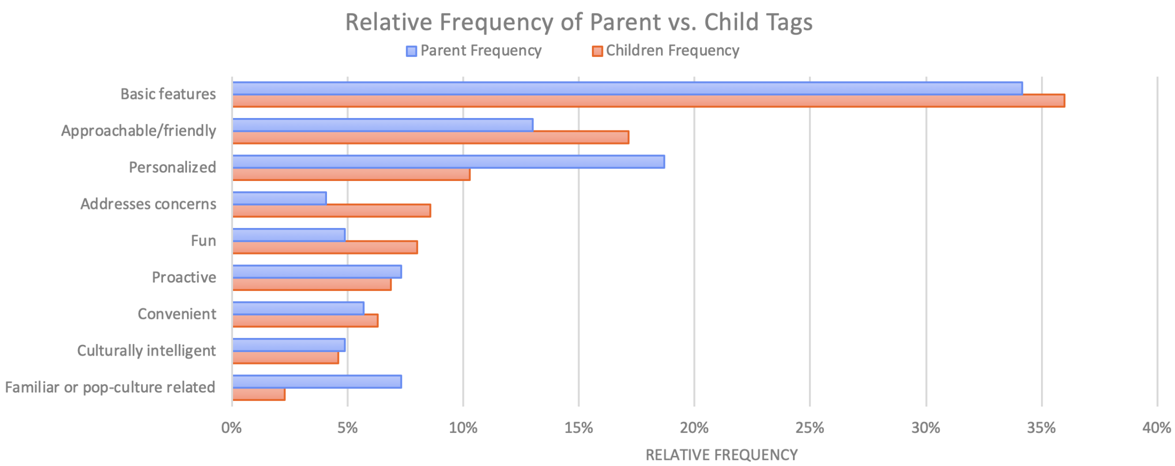


Figure 3-31: Bar charts comparing the frequency of phrases coded with particular tags from children's and parents' responses. The data tagged was from participants' descriptions of their "ideal" agents. To obtain the relative tag frequency, the raw count of each tag was normalized over the total number of tags from that particular subset (e.g., the number of *proactive* tags from children's data was normalized over the total number of tags from children's data). Table 3.15 shows the definitions of each tag.

As shown in Figure 3-31 compared to Figure 3-19, the order of tags from highest to lowest frequency for children and parents was similar to the overall frequency order. The main differences in the children's frequency order included: (1) *addresses concerns* shifting one place more frequent, and (2) *proactive* shifting two places less frequent. The main differences in the parent's frequency order included: (1) *addresses concerns* shifting four places less frequent (to last place), (2) *familiar or pop-culture related* shifting four places more frequent (from last), (3) *personalized* and *approachable/friendly* swapping places, with personalized being more common, and (4) *convenient* and *fun* swapping places, with *convenient* being slightly more common.

Challenging Concepts

There were differences in terms of which concepts children and parents felt were most difficult to learn. Figure 3-32 shows these differences. For example, children more often cited Training and Testing as difficult concepts than parents did; whereas parents more often cited Terminology and Agent Modularization, and more often described other concepts. Otherwise, the relative frequencies were quite similar.

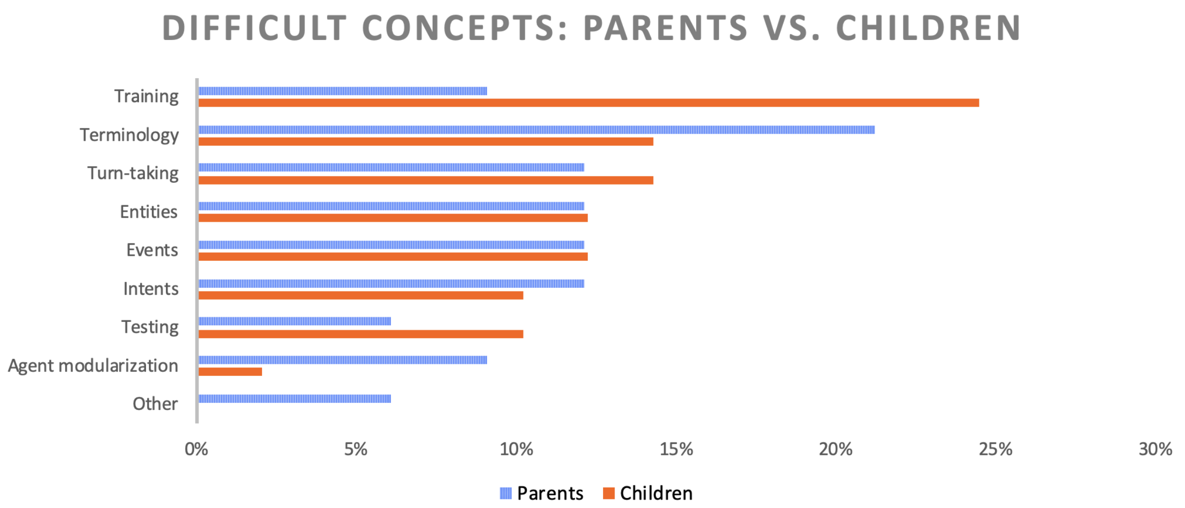


Figure 3-32: Children and parents' responses to the question asking which concepts were most difficult to learn. The graph shows this in terms of relative frequency. Table 3.12 provides descriptions of each concept.

3.4.4 Participants of Different Gender Identities

Demographics and Tutorial Completion

Of the participants who filled out the pre-survey, 20 identified as female, 25 identified as male, and 1 identified as non-binary. Due to an insufficient number of non-binary participants, there is no analysis of this subset of participants. There were no significant differences found between the number of tutorials completed by females vs. males.

Partner Model

Male participants' opinion of Alexa's interactivity (pre/post: $\bar{x}$=2.84,3.42; W(18)=6; p=.024) and companionship (pre/post: $\bar{x}$=3.74,3.26; W(18)=8; p=.039) changed through the workshops, as they felt Alexa was more "start-stop" (less interactive) and more like a friend (less like a co-worker) after the workshops than they did before. There were no significant differences in female participants' opinions overall in terms of the partner model through the workshops.

Before the workshops, females from WEIRD countries thought Alexa was more of a friend than a coworker ($\bar{x}$=3.38,4.27; U(22)=33; p=.011) and less dependable ($\bar{x}$=2.92,3.55; U(22)=44.5; p=.049) than male participants did. Before the workshops, females from non-WEIRD countries thought Alexa was *more* dependable ($\bar{x}$=3.86,2.93; U(19)=23; p=.021) and more competent ($\bar{x}$=1.71,2.93; U(19)=12; p=.002) than male participants from non-WEIRD countries did.

Self-Efficacy and Identity

Female participants saw themselves less as programmers than male participants did before ($\bar{x}$=2.40,3.16; U(43)=174.5; p=.039) and after ($\bar{x}$=3.74,2.77; U(30)=69.5; p=.017) the workshops. However, there was no significant difference directly after the programming activity.

Trust

For those from WEIRD countries, prior to the workshops, female participants thought Alexa reported less correct information than male participants did ($\bar{x}$=3.69,4.18; U(22)=42.5; p=.031) and thought their friends reported more correct information than male participants did ($\bar{x}$=3.31,2.91; U(22)=47; p=.041). These differences were not found for males vs. females from non-WEIRD countries. Figure 3-33 shows the distribution of the results in terms of trust of Alexa's correctness.

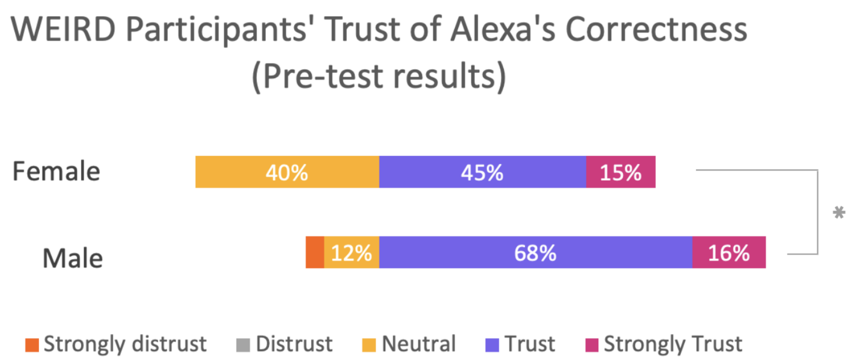


Figure 3-33: Distribution of responses from female and male participants from WEIRD countries when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given before the programming activity.

Challenging Concepts

There were differences in terms of which concepts female and male participants felt were most difficult to learn. Figure 3-34 shows these differences. For example, females more often cited Terminology, Testing and Agent Modularization as difficult concepts than males did; whereas males more often cited Training and more often described other concepts. Otherwise, the relative frequencies were quite similar.

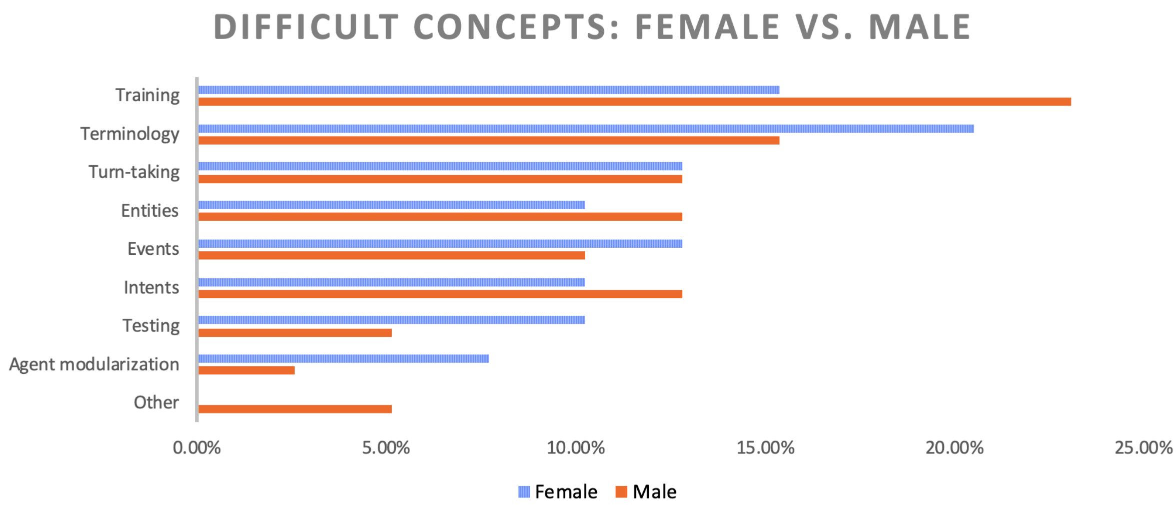


Figure 3-34: Female and male participants' responses to the question asking which concepts were most difficult to learn. The graph shows this in terms of relative frequency. Table 3.12 provides descriptions of each concept.

3.4.5 Participants with Different Levels of Prior Programming Experience

Demographics and Tutorial Completion

In the pre-survey, 14 participants reported they had no prior programming experience, 6 reported they only had only visual (or block-based) programming experience, and 26 reported they had text-based programming experience. (Note that those who had text-based experience may have also had visual programming experience.) There were no significant differences found in the number of tutorials completed depending on the level of prior programming experience participants had.

Partner Model

Prior to the workshops, those who had text-based programming experience thought Alexa was less competent ($\bar{x}$=2.73,2.07; W(16)=0; p=.038) than those who had no programming experience.

Self-Efficacy and Identity

Unsurprisingly, prior to ($\bar{x}$=3.54,1.86; U(38)=62; p=2.50×10^{-4}), during ($\bar{x}$=3.86,2.25; U(31)=46; p=.0011), and after ($\bar{x}$=3.84,2.50; U(25)=30; p=.0065) the workshops, those who had text-based programming experience saw themselves more as programmers than those who had no experience initially. Prior to the workshops, those who had text-based programming experience also saw themselves more as programmers than those who had only visual programming experience ($\bar{x}$=3.54,2.33; U(30)=37; p=.022). This difference was still significant after the programming activity, albeit the difference between means was smaller ($\bar{x}$=3.86,2.83; U(25)=25.5; p=.012). After the entire workshops, however, there was no significant difference between how participants with text-based vs. visual programming experience felt as programmers.

3.4.6 Participants with Different Prior Experience Learning about AI

Demographics and Tutorial Completion

On the pre-survey, 21 participants reported having no prior experience learning about AI, whereas 25 reported they had. There were no significant differences found in the number of tutorials completed depending on whether participants had previously learned about AI or not.

Partner Model

Prior to the workshops, those who had learned about AI previously thought Alexa was more human-like ($\bar{x}$=2.05,2.84; U(44)=-2.87; p=.0063). Participants who had not learned AI before thought Alexa was more dependable after the programming experience (pre/mid: $\bar{x}$=3.47,3.88; W(16)=0; p=.020).

Self-Efficacy and Identity

Those who had learned about AI previously saw themselves more as programmers than those who had not prior to ($\bar{x}$=2.29,3.36; U(44)=-2.73; p=.0092), during ($\bar{x}$=2.82,3.50; U(37)=129; p=.047) and after ($\bar{x}$=3.83,2.87; U(31)=69; p=.0073) the workshops. They were also more confident they could design and create their own technology project than those who did not have prior AI experience prior to ($\bar{x}$=2.71,3.60; U(44)=-2.51; p=.016), during ($\bar{x}$=3.12,4.00; U(37)=92.5; p=.0026), and after ($\bar{x}$=4.28,3.13; U(31)=44.5; $p=2.7\times 10^{-4}$) the workshops. They were also more confident they could make an impact in their community or the world using technology than those who did not have prior AI experience prior to ($\bar{x}$=3.29,4.00; U(44)=-2.52; p=.015), during ($\bar{x}$=3.18,4.18; U(37)=86.5; p=.0015), and after ($\bar{x}$=4.22,3.60; U(31)=80; p=.018) the workshops.

Trust

As shown in Figure 3-35 and 3-36, those who had never learned about AI before thought agents seemed to report correct information more after the programming activity than before (pre/mid: $\bar{x}$=2.88,2.47; W(16)=0; p=.038). No significant difference was found for those who had learned about AI before.

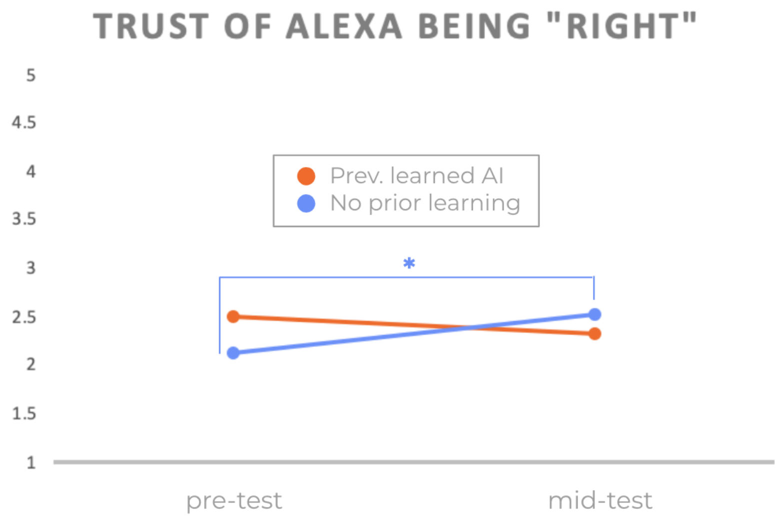


Figure 3-35: Mean responses from participants with no prior experience and prior experience learning about AI when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given before and after the programming activity.

Prior to ($\bar{x}$=3.67,3.00; U(19)=25.5; p=.012) and after ($\bar{x}$=3.80,3.25; U(16)=21; p=.032)

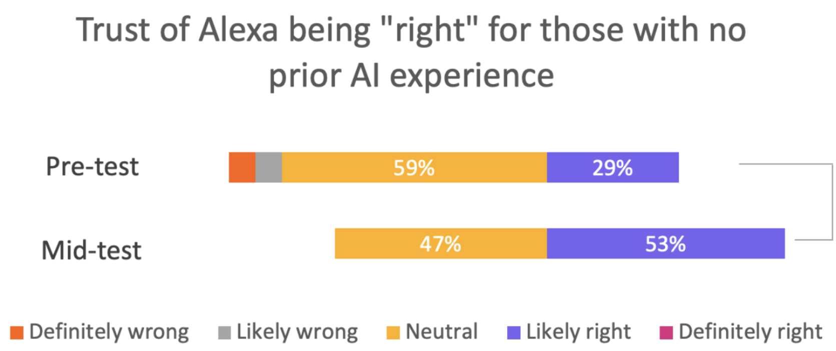


Figure 3-36: Distribution of responses from participants with no prior experience learning about AI when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given before and after the programming activity.

the programming activity, participants from non-WEIRD countries who had *not* learned about AI trusted agents more than those from non-WEIRD countries who had learned about AI. After the societal impact activity, there was no significant difference.

3.4.7 Participants with Different Experiences with Conversational Agents

Demographics and Tutorial Completion

In the pre-survey, 24 participants reported having used more than one type of conversational agent, 16 reported having used only a single type of agent, and 6 reported having never used a conversational agent before. Thirty-eight participants reported typically using conversational agents in their first language and 8 reported typically using them in another language.

There were no significant differences found in the number of tutorials completed depending on whether participants had used more than one type of agent or only a single agent, or on whether participants typically used agents in their first language or not.

Partner Model

Prior to the workshops, those who had experience with more than one agent thought Alexa was more human-like ($\bar{x}$=2.13,2.83; U(38)=120; p=.018) and flexible ($\bar{x}$=2.56,3.25; U(38)=126.5; p=.031) than those who had only experienced a single agent.

Those who used conversational agents in their first language thought Alexa was more human-like prior to ($\bar{x}$=2.61,1.88; U(44)=87.5; p=.025), during ($\bar{x}$=2.66,2.00; U(37)=67;

p=.042) and after ($\bar{x}$=2.82,1.80; U(31)=32; p=.025) the workshops than those who used them in another language. They also thought Alexa was more correct than those who used it in another language, prior to the workshops ($\bar{x}$=4.03,3.00; U(44)=52; p=5.50×10^{-4}).

Self-Efficacy and Identity

Prior to the workshops, those who had experience with more than one agent saw themselves more as programmers ($\bar{x}$=2.19,3.33; U(38)=105; p=.0068), were more confident they could design and create their own technology project ($\bar{x}$=2.50,3.67; U(38)=95.5; p=.0033), and were more confident they could make an impact in their community or the world using technology ($\bar{x}$=3.31,4.00; U(38)=121; p=.021) than those who had only experienced a single agent.

Trust

As shown in Figure 3-37, prior to the workshops, participants who typically used conversational agents in their first language thought Alexa was more correct ($\bar{x}$=4.03,3.00; U(44)=52; p=5.51×10^{-4}) than those who typically used it in another language. There was no significant difference, however, after the programming activity, as shown in Figure 3-38.

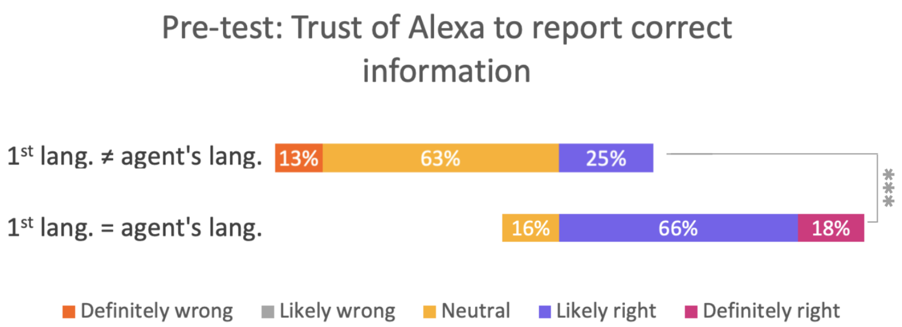


Figure 3-37: Distribution of responses from participants who typically used agents in their first language or not when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given before the programming activity.

3.5 Discussion

This study provided insight into how people of various backgrounds (non-WEIRD and WEIRD, as well as different generations) perceive agents in terms of partner models and

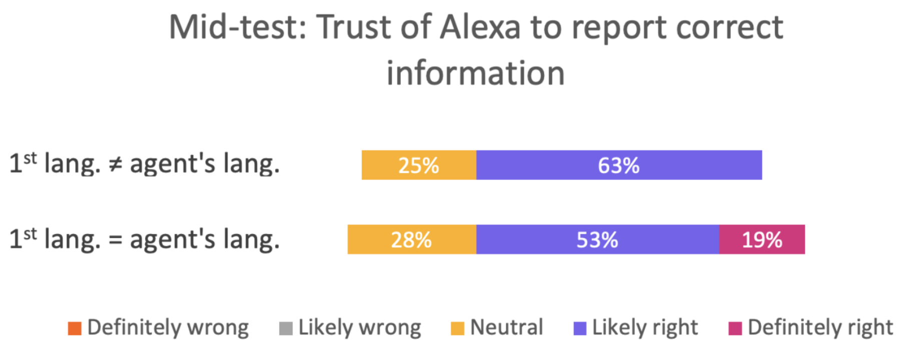


Figure 3-38: Distribution of responses from participants who typically used agents in their first language or not when asked to rate their trust of Alexa's correctness. These results are from 5-point Likert scale question given after the programming activity.

trust, how they learned about agents, their self-efficacy and identification as programmers, and how they envision their ideal agents. There were many significant differences through the workshops when investigating specific subsets of participants. The results led to design recommendations for agent usability ("**DR-U**"), agent pedagogy ("**DR-P**"), and how to align agent development with users' ideal future agents ("**DR-F**"), which are outlined below and summarized in Table 3.2.

3.5.1 RQ3.0.1: Partner Models and Trust

This section discusses how people of various backgrounds perceive Alexa with respect to partner models and trust. It also develops design recommendations for conversational agents (DR-Us) and conversational agent pedagogy (DR-Ps) based on these results.

DR-U1: Inform users about trustworthiness

In general, participants trusted Alexa more than their parents or friends to give correct information. This may indicate an over-trust of Alexa, depending on the actual correctness of the device (although we leave this as a question for future research). Since different agents show varying levels of correctness [101], different agents should be trusted differently. To foster levels of trust matching agents' actual trustworthiness, agents can be designed to indicate their accuracy. For instance, agents may explicitly mention that their information source might be incorrect, or use diction to indicate uncertainty, like "might", "perhaps" or "probably". One real-world example includes how the Google Assistant will often report the source of its information, like how when asking, "Who is the best football player in the world?", it responds with "Here is information from Sporting News", or how Alexa explicitly responds with "Sorry, I don't know that" to the same question. Future areas of research along these lines include determining how such phrases and diction affect users' partner models and feelings of trust towards agents, and quantifying appropriate levels of trust for various agents.

In order to better inform users and allow them to develop more accurate understandings of agents, I propose the design recommendation, **DR-U1: Inform users about trustworthiness**, as shown in Table 3.16. This may include explicit or implicit phrases and diction to indicate how confident the agent (or the agent developer) is about the accuracy or helpfulness of its responses. It also may focus on the aspects of agents' competence, then predictability and then integrity, as participants referenced these aspects most often when describing their trust of agents. More specifically, in their descriptions, participants often referenced whether or not agents understood them correctly, the programming of the agent, and where agents obtained their information; thus, agent designers may want to be transparent about these areas in their designs.

It is important to note too that many people have inaccurate perceptions of agents due to the high accuracy of rule-based computer systems versus the potential inaccuracies or biases of current AI systems. Prior to the proliferation of AI systems (and particularly, machine and deep learning systems), it was fair to assume that computer systems would act very systematically and logically. With current AI systems, however, computer system logic is often obscured by bias [213, 27, 138]. Despite this fact, many participants still indicated they thought AI systems acted without bias. For instance, in response to a question about conversational agents providing correct information, many participants made statements like, "computers are logical beings. They will always say things that are right", and "Conversational Agents act very rational and always speak factually, making them suitable for most situations".

There are similarities between DR-U1 and two of Murad et al.'s design recommendations, G1 (Visibility/Feedback of System Status) and A1 (Ensure Transparency/Privacy). All three recommendations focus on some aspect of informing the user about the agent; however, the particular information each recommendations emphasizes is unique. G1 focuses on informing users about how agents work such that they can effectively complete their goals, A1 focuses on informing users about data collection and their privacy

Table 3.16: Conversational agent usability design recommendations with respect to different subsets of participants' partner models and trust of agents.

Design recommendation for agent usability	**Opportunities in practice**
DR-U1: Inform users about trustworthiness	• Develop conversational agents to include explicit or implicit diction to indicate how confident the agent is about the accuracy of its responses (e.g., "probably", "unsure", "likely not", etc.) • Develop conversational agents with the ability to explain themselves and provide transparency in terms of their abilities to provide trustworthy information [193] • Focus on the competence, then predictability and then integrity aspects of trust when developing agents; for instance, by informing end users how the agent obtains its information, what it does with information given to it and how it is programmed • Since personification may increase users' trust of agents, align the amount of personification of agents with the actual trustworthiness of the conversational agent [193] (see also, DR-U2) • For audiences that may particularly find conversational agents more trustworthy through increased interaction (e.g., children) ensure conversational agent development emphasizes trust indicators throughout interactions • For audiences that may have more initial distrust of agents (e.g., females from WEIRD countries), developers may want to use techniques to encourage trust, such as developing trustworthy personas (see DR-U2)
DR-U2: Design conversational agent personas to foster appropriate partner models	• Design conversational agent personas to foster appropriate relationship building (e.g., whether that means shifting perceptions from co-worker to friend or vice-versa) and therefore trust, as described in [160] (see also, DR-U1) • Develop diverse, multilingual conversational agent personas that many people can relate to and understand (see Section 3.5.1 about similarity and familiarity of conversational agents) • For an audience from non-WEIRD countries, developers may want to focus on creating competent personas • For an adult audience, developers may want to focus on creating more human-like and warm personas • For an audience with lack of trust in agents' correctness, developers may want to ensure their conversational agents use the audience's first language

when using agents, and DR-U1 focuses on informing users about agents' accuracy and trustworthiness.

DR-U1 is also supported by the results from my previous study with ConvoBlocks. In this study, we found correlations between conversational agents' friendliness and participants' perceived trustworthiness, and described the importance of ensuring users' feelings of trust are proportionate to the actual trustworthiness of the device. We also describe how developing conversational agents with the ability to explain themselves, and furthermore, provide transparency in terms of their limitations, may foster an appropriate amount of trust [193].

This trust- and relationship-building effect is not only limited to children working with ConvoBlocks. In this study, we found overall participants' (including parents') feelings shifted towards Alexa being more of a friend than a co-worker through the workshops. Another study found human-agent relationships can be modeled using Knapp's staircase model, which is a model originally developed for human-human relationships [160]. This presents important implications, since relationship building has been linked to increased trust [160, 193], and increased trust has been linked to increased misinformation spread [210, 159]. Thus, it is also important to consider relationship-building in the development of conversational agents. In this study, we investigate human-agent relationships in terms of partner models, as discussed in the next section.

It is also important to consider various audiences and their experiences over time when designing for trust, since different aspects of the workshops affected different subsets of the participants' trust differently. For instance, participants from WEIRD countries' feelings of trust towards Alexa decreased between pre- and post-surveys, but there was no significant differences for participants from non-WEIRD countries in this way. Nonetheless, for the subset of children from non-WEIRD countries, there was a significant difference. Children thought agents were more trustworthy after the programming activity, perhaps because of increased relationship (as described in [160]) or the Dunning-Kruger effect (as described in [193]). In general, children thought Alexa was more correct than parents did after the programming activity, and this perception of correctness increased for children after the societal impact activity. There were also differences in trust for different genders. For instance, female participants from WEIRD countries tended to distrust agents more, and trust their friends more than male participants from WEIRD countries did.

Table 3.16 lists further design implications of these results.

DR-U2: Design conversational agent personas to foster appropriate partner models

At various times in the workshops, there were significant differences between subsets of participants' partner models, or understanding of Alexa as a conversational partner. For instance, after the programming activity, participants from WEIRD countries felt Alexa was less competent, less dependable and more of a peer than an authority figure than those from non-WEIRD countries did. Children felt Alexa was warmer, more human-like, and more dependable than parents did at various times during the workshops. Those with text-based programming experience thought Alexa was less competent than those who had no programming experience prior to the workshops.

Furthermore, subsets of the participants' perceptions of Alexa changed differently through the activities. For instance, before and after the programming activity, children thought Alexa was more human-like than parents did; however, after the societal impact activity, there was no significant difference. After the programming activity, children thought Alexa was more dependable than their parents did, but not before. Male participants' opinion of Alexa's interactivity and companionship changed through the workshops; however, there were no significant differences in female participants' opinions overall in terms of the partner model through the workshops.

Thus, different activities may change certain people's perceptions of conversational agent partner models, but not others'; and certain people may have particular initial perceptions that others do not share. This could have implications for how people of different backgrounds and ages interact with Alexa. According to Hinds et al., team members or partners are chosen according to their reputation for being competent, strength of prior relationship, and perceived similarity [74]. In this context, users may interact with agents as conversational partners more often if the agents are perceived as competent, friends, and similar to users. This leads to another recommendation, **DR-U2: Design conversational agent personas to foster appropriate partner models**, as described in Table 3.16.

This recommendation could vary greatly depending on the intended agent audience. For instance, since participants from WEIRD countries felt Alexa was less competent and dependable than those from non-WEIRD countries, perhaps agent designs for audiences in non-WEIRD countries should focus on increasing this perception of competence. Since those who used agents in their first language thought Alexa was more human-like before, during and after the workshops than those who used them in another language, and Hinds et al. note that partners are often chosen based on similarity or familiarity, perhaps agent designers should focus on developing multilingual and diverse agent personas depending on their intended audience. Table 3.16 includes design implications, such as these, with respect to different audiences.

DR-P1: Encourage trust of pedagogical agents (to the extent of their trustworthiness) to facilitate learning from them

There are also implications for conversational agent pedagogy when considering the trustworthiness of agents. For instance, to facilitate student learning when agents are acting as teachers or mentors, it is important the student trusts the agent [193, 155, 2, 24]. Thus, I propose the design recommendation, **DR-P1: Encourage trust of pedagogical agents (to the extent of their trustworthiness) to facilitate learning from them**.

Considering the correlations between personification of agents and trust [160, 193], it may be helpful for agents in teaching roles to be personified, while balancing the potential for over-trust by considering DR-U1. This trust-building may be encouraged by teachers, or through increased interactions with the agents. One example supporting this idea is how participants overall shifted their perceptions of Alexa towards being more of a friend through interacting more with Alexa and going through the workshops. Such increased feelings of friendship may also increase feelings of trust long-term [5, 160].

When describing how their trust changed through the programming activity, participants most often referenced predictability, then competence and then integrity. They also emphasized how learning about agents, including learning about how agents are programmed, agents' sources of information and how agents understand information given to them, affected their sense of trust. Thus, in conversational agent educational activities, educators may want to emphasize these concepts to encourage appropriate levels of trust. Table 3.17 includes these ideas.

DR-P2: Engage students in activities that will reinforce the concepts being taught with respect to their conversational agent partner models

The results from this study regarding partner models also have implications for conversational agent pedagogy, especially since different activities seemed to change different groups of participants' perceptions of agents differently. Thus, I propose the design recommendation, **DR-P2: Engage students in activities that will reinforce the concepts being taught with respect to their conversational agent partner models**. For example, to reinforce how conversational agent technology is still in its infancy (and agents are not highly competent in all tasks), for a WEIRD audience, a programming activity may be appropriate, but for a non-WEIRD audience, a more direct instruction approach may be appropriate. To level-set perception of agent competence between those with and without text-based programming experience, a visual programming tutorial on agent development may be appropriate. To increase perceptions of agent dependability for those who have not learned about AI before, a programming activity may be appropriate. To increase perceptions of agent human-likeness, using diverse, relatable agents and agents in the audience's first language may be appropriate. Table 3.17 outlines these implications.

Table 3.17: Conversational agent pedagogy design recommendations with respect to different subsets of participants' partner models and trust of agents, what they found most difficult, and their self-efficacy and identification with being a programmer when learning about and creating agents.

Design recommendation for conversational agent pedagogy	Opportunities in practice
DR-P1: Encourage trust of pedagogical agents (to the extent of their trustworthiness) to facilitate learning from them	• Encourage student trust of pedagogical agents through enabling students to interact with the agents more often and in ways that encourage friendship-building • Encourage student trust of pedagogical agents through teaching students how agents work • Use personified or friendly pedagogical agents • Emphasize the aspects of predictability, then competence and then integrity when teaching about agents' trustworthiness; for instance, by teaching about how agents are programmed, agents' sources of information and how agents understand information given to them

DR-P2: Engage students in activities that will reinforce the concepts being taught with respect to their conversational agent partner models	• To reinforce how agents are not highly competent in all tasks, for a WEIRD audience, a programming activity may be appropriate, but for a non-WEIRD audience, a more direct instruction approach may be appropriate • To level-set perception of agent competence between those with and without text-based programming experience, a visual programming tutorial on agent development may be appropriate • To increase perceptions of agent dependability for those who have not learned about AI before, a programming activity may be appropriate • To increase perceptions of conversational agent human-likeness, using diverse agents and agents in the audience's first language may be appropriate • To increase children's feelings of trust of conversational agents' correctness, engaging them in programming activities may be appropriate
DR-P3: Teach visual programming to empower nearly anyone to program conversational agents	• Use environments like *MIT App Inventor* [186], *Scratch* [157], or *Snap!* [83] to simplify programming syntax and teach complex conversational agent concepts to students with different prior experience
DR-P4: Emphasize concepts that are challenging for particular audiences	• Emphasize Training, Turn-taking, Machine learning, Societal impact and ethics, Constrained vs. unconstrained natural language, and general terminology when teaching conversational agent curricula • When teaching children, emphasize Training and Testing • When teaching parents, emphasize Terminology and Agent Modularization • When teaching those from non-WEIRD countries, emphasize Training and Events • When teaching those from WEIRD countries, emphasize Testing and Turn-taking
DR-P5: Include both societal impact and conversational agent development activities to facilitate computational action	• Including societal impact activities can increase audience members' confidence in making an impact in their community or the world using technology • Examples of societal impact curricula include the discussion activities in the Zhorai, Convo and ConvoBlocks curricula [99, 215, 186], and the technology re-design and ethical matrix activities in Payne's and Ali et al.'s curricula [6, 138] • Including programming activities can be especially important to encourage children females' self-efficacy and identities as programmers
DR-P6: Encourage and provide consistent programming opportunities to underrepresented minorities	• Providing additional opportunities to children from non-WEIRD countries may enable them to obtain the same benefits in terms of self-efficacy and identity as programmers as those from WEIRD countries • Providing more consistent programming opportunities may enable females to sustain the benefits they gain in terms of self-efficacy and identity as programmers from such opportunities

DR-P7: Supplement conversational agent development activities with additional agent engagement, AI learning and programming activities	• Encouraging activities such as AI learning experiences, experiences with more types of conversational agents and programming, in addition to conversational agent curricula may increase self-efficacy and identification as programmers

In terms of trust, those who had never learned about AI before thought agents seemed to report more correct information after the programming activity. Thus, for those with prior knowledge of AI, programming activities might be beneficial in terms of learning to trust pedagogical agents more. Furthermore, those who used Alexa in their first language trusted Alexa to give correct information more than those who used it in another language prior to the workshops. After the programming and societal impact activities, however, there were no significant differences. Thus, for those who are not using conversational agents in their first language, more exposure to agents and programming activities may contribute to increased trust. These examples illustrate how people of various backgrounds' partner models and feelings of trust towards agents might shift, and how DR-P2 may be implemented in practice, as shown in Table 3.17.

3.5.2 RQ3.0.2: Difficulties Learning about Conversational Agents

This section discusses what people of various backgrounds find most difficult when learning about and creating conversational agents. It also develops design recommendations for conversational agent pedagogy (DR-Ps) based on these results.

DR-P3: Teach visual programming to empower nearly anyone to program conversational agents

In terms of the number of tutorials completed, there were no significant differences between male and female, non-WEIRD and WEIRD, and parents and children. This indicates all the major subsets of participants are equally capable of learning to program conversational agents with ConvoBlocks, despite potentially having differing levels of prior programming, conversational agent or AI experience. For example, children had more prior experience programming than parents, and participants from non-WEIRD countries had experienced fewer types of agents than those from WEIRD countries. Furthermore, there were no significant differences in terms of tutorial completion for those with different prior levels of programming experience, different amounts of experience learning AI, and who had used more than one or a single conversational agent. Thus, I propose the design recommendation, **DR-P3: Teach visual programming to empower nearly anyone to program conversational agents**. This aligns with other research in which complex concepts and technology development are taught through visual programming, including app, IoT, and AI development [83, 209, 143]. It is also supported by how there was no significant difference in terms of programming identity between those who had prior text-based vs. visual programming experience after the workshops, as discussed in Section 3.5.3.

DR-P4: Emphasize concepts that are challenging for particular audiences

In general, the top three concepts most referenced as difficult in this study included Training, Terminology and Turn-taking. Although this trend remained relatively similar for all major subsets, different subsets of the participants still found different concepts more challenging than others. For example, children cited Training and Testing more often as difficult concepts than parents did; whereas parents cited Terminology and Agent Modularization more often than children did. Participants from non-WEIRD countries cited Training and Events more often as difficult concepts than those from WEIRD countries did; whereas participants from WEIRD countries cited Testing and Turn-taking more often than those from non-WEIRD countries did. In other studies, students found "Machine learning" and "Societal impact and ethics" [191], and "Constrained vs. unconstrained natural language" [215] particularly challenging to learn. Educators may want to focus on particularly challenging concepts for their students; thus, I propose the recommendation, **DR-P4: Emphasize concepts that are challenging for particular audiences**.

3.5.3 RQ3.0.3: Self-Efficacy and Identity as Programmers

This section discusses people of various backgrounds' self-efficacy and identification with being a programmer before, after and while creating conversational agents. It also develops recommendations for conversational agent pedagogy (DR-Ps) based on these results.

DR-P5: Include both societal impact and conversational agent development activities to facilitate computational action

Overall, participants' confidence in feeling like programmers and creating their own technology projects significantly increased through the programming activity as well as the societal impact activity. Participants' confidence in terms of being able to make an impact in their community or the world, however, only significantly increased through the societal impact activity. As described in Section 1.2.1, this work seeks not only to empower young people to understand and develop their own conversational agents, but also to empower them to take computational action—or have authentic impact through programming—in their communities [176]. These results align with other CS and AI education studies [176, 177, 95, 134], and strongly support the inclusion of both conversational agent development activities and societal impact activities in pedagogy to foster computational action. Thus, I propose the following pedagogical recommendation, **DR-P5: Include both societal impact and conversational agent development activities to facilitate computational action**.

DR-P6: Encourage and provide consistent programming opportunities to underrepresented minorities

Despite how the activities increased overall participants' self-efficacy and identification with being programmers, certain subsets of the participants did not benefit as much as others. (This is despite being equally capable of creating conversational agents, as discussed in Section 3.5.2.) For example, children from WEIRD countries felt more confident they could make an impact in their community or the world after the programming activity and societal impact activity than children from non-WEIRD countries. Thus, the programming and societal impact activities seemingly influenced children from WEIRD countries more than children from non-WEIRD countries. Before and after the entire workshops, female participants identified significantly less as programmers than male participants. However, directly after the programming activity, there was no significant difference between females' and males' self-efficacy and identities as programmers. Females having lower self-efficacy scores is consistent with other computer science literature [17, 84], and might mean giving females more opportunities to engage with programming activities in order to maintain self-efficacy and identity gains. Thus, I propose the pedagogical DR, **DR-P6: Encourage and provide consistent programming opportunities to underrepresented minorities**, specifically those from non-WEIRD countries and those of underrepresented genders.

DR-P7: Supplement conversational agent development activities with additional agent engagement, AI learning and programming activities

Furthermore, those who had more background and experience in related topics generally had increased self-efficacy and identified more as programmers. For instance, those who had learned about AI previously identified more as being programmers, being able to design and create their own technology projects and being able to make impacts in their communities or the world using technology throughout the experience. Those who had experience with more than one agent (as opposed to those who had only experienced a single agent) felt the same type of increases as those who had learned AI. Those who had text-based programming experience saw themselves more as programmers than those with no prior experience throughout the entire experience. For those with initial visual programming experience, however, after engaging with the conversational agent workshops, there was no significant difference in terms of programming identity with those who had text-based experience. Thus, creating meaningful visual programming projects and engaging in societal impact curricula may impact those with some visual programming experience more than those without any. Thus, I propose the design recommendation, **DR-P7: Supplement conversational agent development activities with additional agent engagement, AI learning and programming activities**, as these opportunities likely have significant impact on people's identity and self-efficacy as programmers. Also note, however, how these prior experiences (with additional conversational agents, with AI learning activities and with programming) may indicate differences in socioeconomic class, so the experiences themselves may not have caused the benefits seen in this study. Nonetheless, providing more opportunities for these type of activities to diverse audiences is key to democratizing technology.

3.5.4 RQ3.0.4: Envisioning Future Conversational Agents

This section discusses how people of various backgrounds envision their ideal conversational agents. It also develops recommendations for aligning conversational agents with people's ideal future conversational agents (DR-Fs) based on these results.

DR-F1: Design with more task-orientation in general, while considering the end user audience

All major subsets of participants (i.e., children, parents, participants from non-WEIRD countries and participants from WEIRD countries) and combinations of these subsets (e.g., children from WEIRD countries) described their ideal agents with more of a task orientation rather than a social orientation. Example task and non-task oriented phrases can be found in Table 3.13. Overall, participants described their ideal agents with 75% task-oriented and 25% non-task oriented attributes. This may be explained by the influence of current agents on participants' responses, since current agents tend to be task-oriented, rather than truly conversational or social [38]. That being said, participants also included non-task oriented agent attributes in their responses, like having agents ask about how users are feeling, asking about users' days, or telling users the agents' feelings. There were also different ratios of task vs. non-task orientations depending on the subset. For instance, children had a non-task orientation of 25%, whereas parents had a non-task orientation of 18%; and participants from non-WEIRD countries had a non-task orientation of 30% whereas participants from WEIRD countries had a non-task orientation of 20%. Thus, I suggest considering developing agents with varying amounts of task-orientation with respect to the intended users, **DR-F1: Design with more task-orientation in general, while considering the end user audience**. This DR is explored with respect to each of the major subsets of participants in Table 3.18.

DR-F2: Balance personification and artificiality in agent design while considering the end user audience

Participants also commented on agents' human-likeness and artificiality, with statements such as "[My ideal agent would be] like a robot, but not human like otherwise it would be a bit creepy" and "[My ideal agent] looks or sounds human, or has human-like emotions", as shown in Table 3.14. Overall, they described their ideal conversational agents with a slight preference for human-like agents over artificial ones. This varied greatly between the major subsets of participants, as shown in Figure 3-18. For instance, children had a slight preference for artificial agents (52% of comments), whereas parents had a large preference for human-like agents (70% of comments). Similarly, those from non-WEIRD countries had a slight preference for artificial agents (55% of comments), whereas those from WEIRD countries had a large preference for human-like agents (73% of comments). Thus, I propose the design recommendation, **DR-F2: Balance personification and artificiality in agent design while considering the end user audience**, as shown in Table 3.18. This aligns with human-robot interaction research around the "uncanny valley", which describes how—for humans to have an affinity for a robot—it is undesirable for a system to be completely human-like or completely artificial [114].

DR-F3: Focus development on useful, common features; user-orientation; enjoyable interactions; and emotional intelligence, while emphasizing certain aspects depending on the end user audience

There were nine themes other than task-orientation and human-likeness, which are outlined with examples in Table 3.15. Three of the themes indicate participants want future conversational agents to be user-oriented, including *Convenient*, *Personalized*, and *Proactive*; two indicate a desire for enjoyable interactions, including *Approachable/friendly*, *Familiar or pop-culture related*, and *Fun*; and three others indicate a desire for emotional intelligence, including *Addresses concerns* and *Culturally intelligent*. The final theme, *Basic features*, indicates participants want future agents to include the typical features current agents have, like the ability to play music or get the weather.

In terms of relative importance to participants overall, *Basic features* were referenced most often, with 105 tagged phrases; user-oriented phrases were tagged 80 times; phrases related to enjoyable interactions were tagged 79 times; and phrases related to emotional intelligence were tagged 34 times. There were also some notable differences in terms of relative importance of certain themes between subsets. For instance, in terms of theme rankings, both parents and those from WEIRD countries emphasized pop-culture references while de-emphasizing addressing concerns. This was also true in terms of the comparisons shown in Figure 3-26. This figure also shows how those from non-WEIRD countries emphasized cultural intelligence with respect to the results from those from WEIRD countries, how parents emphasized personalization with respect to the results from children, and how children emphasized friendliness and approachability with respect to the results from parents. Thus, I propose the design recommendation, **DR-F3:**

Focus development on useful, common features; user-orientation; enjoyable interactions; and emotional intelligence, while emphasizing certain aspects depending on the end user audience, as described in Table 3.18. This is also supported by the results from another study with ConvoBlocks, in which students reported most often using basic, useful features when interacting with Alexa [193].

Table 3.18: Design recommendations for future conversational agents with respect to how different subsets of participants described their ideal conversational agents.

Design recommendation for the future of conversational agents	Subset-specific information
DR-F1: Design with more task-orientation in general, while considering the end user audience	**Children:** Relatively more non-task (social) orientation (29%) than overall participants **Parents:** Relatively less non-task (social) orientation (18%) than overall participants **Non-WEIRD:** Relatively more non-task (social) orientation (30%) than overall participants **WEIRD:** Relatively less non-task (social) orientation (20%) than overall participants
DR-F2: Balance personification and artificiality in agent design while considering the end user audience	**Children:** Larger preference for artificial (52%) than overall participants **Parents:** Larger preference for human-likeness (70%) than overall participants **Non-WEIRD:** Larger preference for artificial (55%) than overall participants **WEIRD:** Larger preference for human-likeness (73%) than overall participants
DR-F3: Focus development on useful, common features; user-orientation; enjoyable interactions; and emotional intelligence, while emphasizing certain aspects depending on the end user audience	**Children:** More emphasis on the *Addresses concerns* and *Approachable/friendly* themes; and less emphasis on the *Familiar or pop-culture* and *Personalized* themes **Parents:** More emphasis on the *Familiar or pop-culture* and *Personalized* themes; and less on the *Addresses concerns* and *Approachable/friendly* themes **Non-WEIRD:** More emphasis on the *Addresses concerns* and *Culturally intelligent* themes; and less emphasis on the *Familiar or pop-culture* and *Basic features* themes **WEIRD:** More emphasis on the *Familiar or pop-culture* and *Basic features* theme; and less on the *Addresses concerns* and *Culturally intelligent* themes

Table 3.19: Description of the themes in participants' answers to the questions, "Please explain why you think conversational agents say things that are right/wrong", "Do you think you changed your opinion on whether conversational agents say things that are right/wrong since going through the activities? [Why?]", and "Do you think you changed your opinions on any of the above [partner model] questions since going through the activities? [Why?]".

Category	Tag	Definition	Example utterances
Opinion changed	Opinion did change	Through the activities, the participant's perception of their partner model or trust of correctness changed	"Yes definitely, the workshop has changed it in many different all amazing ways", "I think that my opinion has changed a little"
	Opinion did not change	Through the activities, the participant's perception of their partner model or trust of correctness of agents did not change	"No, I have not", "not much has changed in my opinion", "My opinion has not changed"
	Ambiguous	It is not clear whether the participant's opinion changed or did not change	"Tried to run Alexa and result in utterances", "I don't trust them very much because I am skeptical of what they do with our information", "It is all the 'man behind the gun' "
Trustworthiness	Trustworthy	The agent provides reasonable and accurate information	"because it can give information almost 100% true", "I think they say things that are correct", "Again a computer is logical the things that they can say are logical and mostly correct when it regards facts"
	Not Trustworthy	The agent gives incorrect or unreliable information	"they can get sources but they don't know if it's reliable", "Conversational agents sometimes say things wrong as they collect data from multiple sources, which can often contradict each other"
	Complex	In between trustworthy and not: agent can be trustworthy or untrustworthy	"Conversational agents often pull information from online sources, which could possibly be correct but could also be wrong as information from certain sources changes in both what the information contains and its accuracy", "I think it depends on the program itself"
	Unsure	No opinion or not enough information to form an opinion	"I don't have much experiences with conversational agents so i'm not sure myself", "I have no opinion"

What changed (partner model)	Machine-like/ human-like	Addresses the artificiality or human-likeness of the agent	"It does have potential to be more human like.. but there is still a long way to go", "The fact that behind alexa human like voices are just programs by people Alexa can be trained to do better"
	Inflexible/ flexible	Addresses the flexibility of the agent	"I've learned that Alexa can be a bit more flexible than what I've experienced", "I would think it sort of showed me in a sense the inflexibility of Alexa"
	Unreliable/ dependable	Addresses the dependability of the agent	"I am more confident in the reliability of Alexa's understanding of questions", "i Think alexa is a bit more dependable then when i start this activety"
	Interactive/ start-stop	Addresses the interactivity of the agent	"I did change one of my opinions because I found out that alexa sometimes isn't really that start-stop", "Yes I think as we progressed I could see ways for more interaction and personalized connections with the device"
	Competent/ incompetent	Addresses the competence of the agent	"it increased my feelings about its competence", "Became more knowledgeable and aware of Alexa competencies"
	Authority/ peer	Addresses the authority of the agent	"Now I think Alexa is more like peer rather than a authority figure", "Alexa was never an authority figure—if it was replaced with 'servant', I might have put it closer to the middle"

Reasoning	Agent-related reasoning: Programmed	Programmed by humans, programmed to know things, program has bugs/errors	“The fact that behind alexa human like voices are just programs by people Alexa can be trained to do better”, “Because the conversational agents are programmed to give you correct information”, “they don’t know everything unless they are programmed to know it”
	Agent-related reasoning: Experience with agent	The participant had impactful interactions with conversational agents	“I think that I’ve learned that Alexa can be a bit more flexible than what I’ve experienced”, “I would think it sort of showed me in a sense the inflexibility of Alexa, as when coding it”
	Agent-related reasoning: Nature of the agent	Agent is factual, agent generally does not understand, agent has limited knowledge, etc.	“conversational AI are still new technology that are still being improved”, “they mispronounce things”, “Conversational Agents act very rational and always speak factually, making them suitable for most situations”
	Agent’s data source: Internet	Agent draws information from internet sources, internet information has varying degrees of accuracy, etc.	“no, because their data is still from the internet”, “the information on the Internet to provide answers, which could be right or wrong”, “Based on Google search results”
	Agent’s data source: Human data	Agent uses data from humans to formulate responses, agent is trained based on data it is given, etc.	“conversational AI’s sources are from humans so there will always be a mistake that the conversational agents will make”, “They take their information from things we have set ourselves”, “learns from it so depends on how it is trained”, “It all depends on how’s it’s been trained”
	Agent’s data source: Undisclosed	Agent obtains information, but it is unclear where the information is from	“the information we get from the agents are right or wrong depends on the sources of the agents which could be incorrect”, “they can get sources but they don’t know if it’s reliable”, “not all information that are publicly available are accurate”
	Personal reasoning: Personal learning	The learning might be implied, like how the workshops were useful or how they did not know much before	“yes, because i learned something from this zoom meeting”, “I do think that after learning, it does help me realize the feature and the potential that these agents have”, “But i have learned more about it so i have different opinions about it”
	Information given to agent: User input	The way the user phrases the question, accent, etc.	“it really depends on the situation and how complex the question was”, “Always room for misinterpretation”, “depend on the input of the user”, “can’t understand accent”
	Information given to agent: Data disclosure	What happens with the information, government spying, etc.	“because someone or even goverment can spying on us”, “I am skeptical of what they do with our information”, “we really have no way of confirming what a personal assistant does with the information it collects from you”
	Ambiguous	No reason given or not explained well	“I changed a few opinions”, “No”, “No I think I have stayed the same”, “It does not always give you the correct answer”

3.6 Limitations and Future Work

This study has four main limitations and areas for future work. The first is the demographics of the participants. Although we had participants from around the world, there were large groups of participants from Indonesia in the non-WEIRD category and the United States in the WEIRD category. Furthermore, there were more child than parent participants, meaning the overall results may be skewed towards the children's results. Future studies could incorporate more participants and further balance them across subsets to verify our results.

A second limitation is how we only utilized one type of agent (i.e., Alexa) in this study. This agent has a default female voice, which could affect participants' views on partner models and trust. Future studies could incorporate various types of agents and compare participants' views of each type.

A third limitation is the context of our study. Specifically, we focused on trust of agents in an educational setting; however, future studies could investigate trust in other contexts. Furthermore, future studies could investigate how levels of trust affect how people act, since Gaube et al. found levels of trust do not necessarily correspond to trusting actions—at least in clinical settings [59].

A final limitation is how we did not investigate what "healthy" or "appropriate" levels of trust or partner models are for agents. For example, future research could investigate the trustworthiness of various agents' information, determine people's level of trust towards these agents, and how the agents and their information affect people's actions. The results from this study could help educators and designers foster healthy levels of trust in end users.

Chapter 4

Analysis of Conversational Agent Development Platforms and Learning Activities

In many schools today, the phrase 'computer-aided instruction' means making the computer teach the child. One might say the computer is being used to program the child. In my vision, the child programs the computer and, in doing so, both acquires a sense of mastery over a piece of the most modern and powerful technology and establishes an intimate contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building.
—Papert [136]

In order for K-12 students to understand the trustworthiness of different conversational agents, it is important for them to understand how these agents work. Despite much research in related subfields, including improving conversational AI technology, developing pedagogical agents, and teaching AI concepts; currently, there is little-to-no published research on how to teach people conversational agent concepts and development techniques—especially at the K-12 level [142, 178, 100, 33, 7]. In this chapter, I analyze conversational agent development platforms and learning activities to develop a comprehensive framework of conversational agent concepts. These concepts can be used as guides to develop conversational agent curricula, curate future research in conversational AI pedagogy, and as a starting point for developers beginning to create their own agents.

To develop this conversational agent concept framework, I analyzed development platforms and activities with a broad scope in terms of the target audience's prior programming and AI experience. I first analyzed platforms I developed alongside others at MIT, including Zhorai [99], ConvoBlocks [185], and Convo [190]. Each of these platforms have associated studies in which K-12 students learn conversational agent concepts. I then analyzed conversational agent platforms and activities from industry. The purpose of the analysis was to identify (1) the conversational agent development concepts taught, (2) how (or whether) they taught the prominent conversational agent design recommendations from Chapter 2 (or the additional design recommendations from Chapter 3), and (3) the complexity of the platform designs and whether they could apply to the K-12 context. In Chapter 5, I outline how to teach the conversational agent concepts identified in this chapter effectively in a K-12 context, and ultimately answer the research question, **RQ4.0.1: How can we teach complex conversational agent concepts and design recommendations found in prominent platforms to K-12 students?**

Table 4.1 lists the concepts from the analysis with respect to which activities first introduced them or significantly developed them. The Conversational AI for Kids research team at MIT reviewed these concepts in terms of their ability to comprehensively support students' understanding of conversational agents. I organized the concepts into five categories: (1) **Natural Language Understanding**, (2) **Conversation Representation**, (3) **Dialog Management**, (4) **Data Access and Conversation Context**, and (5) **Human-Agent Interaction**. I derived these categories from conversational agent technology research [142, 33, 7], AI education research [178, 100], and the concepts taught through the platforms and related activities described below [99, 185, 215, 14, 173].

Table 4.1: A summary of which activities first introduced (■) and significantly developed (⊡) which conversational agent concepts.

	Concept	**Zhorai**	**Convo-Blocks**	**Convo**	**Alexa Devl. Console**	**Dialog-flow**
Natural Language Understanding	Semantic analysis	■				
	Machine learning	■				
	Similarity scores	■				
	Large language models		■			
	Transfer learning		■			
	Intents		■			
	Entities		■			
	Training		■		⊡	⊡
	Testing		■			
	Constrained vs. unconstrained natural language			■		⊡
Conversation Representation	Textual representations	■				
	Undirected graphs	■				
	Histograms	■				
	Event-driven program representations		■		⊡	
	Storyboards				■	
	Directed graphs					■
Dialog Management	Turn-taking	■				
	Events		■			
	Entity-filling		■		⊡	
	Conditions		■			⊡
	Conversation State			■		
	State machines					■

Data Access and Conversation Context	Pre-programmed data	■				
	User-defined data	■				
	Contextual data		■		⊡	
	Agent modularization		■			
	Device access		■		⊡	
	Cloud computing		■	⊡		⊡
	Webhooks and APIs				■	
	Flow and page modularization					■
Human-Agent Interaction	Speech synthesis	■	⊡			
	Speech recognition	■	⊡		⊡	
	Recovery	■				
	Societal impact and ethics	■	⊡			
	Text-based interaction		■	⊡		
	Voice-based interaction		■	⊡		
	Multimodal interaction		■		⊡	
	Task- vs. non-task-oriented			■		
	Deployment				■	⊡
	Effective conversation design				■	⊡

In addition to addressing the concepts developed in this chapter, the activities addressed the design recommendations from Chapter 2 and 3. I show which activities addressed which of the recommendations in Table 4.2. As discussed throughout the analysis, the activities addressed these recommendations in different ways. Some addressed them through explicitly describing the recommendations or teaching students to create agents using these recommendations (as shown with the filled square in Table 4.2). Others addressed them implicitly through engaging students with an agent that had been designed according to these recommendations (as shown with the empty square in Table 4.2). Still, others addressed them in extraneous materials, like the Dialogflow Voice Design Best Practices [175], which were only referenced in the activity materials (as shown with the square with a dot inside in Table 4.2).

Table 4.2: This table outlines which design recommendations were implicitly and explicitly addressed by each agent activity. A filled square (■) represents addressing a recommendation directly in the materials, an empty square (□) represents addressing a recommendation indirectly in the materials (e.g., the Zhorai agent using the principles in conversation with students), and a square with a dot inside (⊡) represents addressing the recommendation in related materials external to the activity (e.g., Google's Dialogflow Voice Design Best Practices [175]). The external materials are cited next to the symbol.

Design Recommendation (see Chapters 2 and 3)	**Activity 1: Zhorai**	**Activity 2: Convo-Blocks**	**Activity 3: CONVO**	**Activity 4: Alexa Devl. Console**	**Activity 5: Dialogflow**
G1: Visibility/Feedback of System Status	□	■	□	■	⊡ [174, 175]
G2: Mapping Between System and Real World			□	■	⊡ [175]
G3: User Control and Freedom	□	■	■	■	
G4: Consistency throughout the Interface			□	■	
G5: Preventing User Errors		■	■	■	⊡ [174, 175]
G6: Recognition Rather than Recall	□		□	■	⊡ [174]
G7: Flexibility and Efficiency of Use	□	■	■	■	⊡ [175]
G8: Minimalism in Design and Dialogue	□		□	■	
G9: Allowing Users to Recognize and Recover from Errors	□	■	□	■	⊡ [174, 175]
G10: Providing Help and Documentation	□		□	■	⊡ [174, 175]
A1: Ensure Transparency/Privacy				■	

A2: Considering How Context Affects Speech Interaction			□	■	
DR-U1: Inform users about trustworthiness					⊡ [175]
DR-U2: Design conversational agent personas to foster appropriate partner models	□			■	

The platforms and activities are as follows.

1. Learning Activity 1 (Novice): This activity investigates how conversational agents can learn through classifying animals into ecosystems with the agent, **Zhorai** [99] (Section 4.1).
2. Learning Activity 2 (Beginner): In this activity, students develop an intent- and entity-enabled conversational agent with **ConvoBlocks** through block-based coding and training conversational agents [185] (Section 4.2).
3. Learning Activity 3 (Intermediate): Students develop an intent- and entity-enabled conversational agent with **Convo** through natural language programming and training conversational agents [215] (Section 4.3).
4. Learning Activity 4 (Advanced): Deverlopers create an intent- and entity-enabled conversational agent with the **Alexa Developer Console** through programming in JavaScript and training conversational agents [14] (Section 4.4).
5. Learning Activity 5 (Expert): Developers create an intent-, entity-, and state-machine-enabled conversational agent with **Dialogflow CX** through developing conversation flows and pages, and training conversational agents [173] (Section 4.5).

The first three activities—the Zhorai, ConvoBlocks and Convo activities—involve platforms I developed for this dissertation. Each of them target students in K-12 education. The fourth and fifth activities—the Alexa Developer Console and Dialogflow CX activities—are industry-based. They target developers with greater prior experience programming and knowledge of conversational agents. These activities teach concepts, like *webhooks and APIs* and *state machines*, which are not included in the K-12 activities; however, could be useful for young developers to encounter earlier in their learning.

There are other industry-based conversational agent development platforms available, like Alexa Blueprints [9] and SiriKit [15]; however, I chose the particular activities above as they target developers with a range in prior experience and provide associated educational materials. Other tools and agents were not chosen due to their lack of educational materials associated with conversational agents (e.g., Alexa Blueprints [9]), limitations in scope (e.g., SiriKit [45, 22, 73]), or lack of prominence (e.g., promptly-dotnet [106]). As the conversational agent development landscape changes, there will likely be additional concepts and design recommendations to teach students. Nonetheless, this framework provides an initial step towards teaching K-12 students currently-held conversational agent concepts comprehensively.

The following sections describe the platforms and related educational materials, and analyze the conversational agent concepts and design guidelines they teach. Additionally, teaching materials for the Zhorai, ConvoBlocks and Convo activities can be found in Appendix E, F and G, respectively. Each of their code repositories can be found on GitHub [192, 204, 194].

4.1 Activity 1: Zhorai

The first activity, Zhorai, exposes primary school-aged children to conversational agents while teaching them machine learning concepts. In this section, I describe the platform (as shown in Figure 4-1) and embedded educational materials. Much of this information, as well as results from a study with Zhorai and children in 3-5th grade can be found in our paper [99]; teaching resources for the Zhorai activity can be found in Appendix E; and the Zhorai code repository can be found on GitHub [192].

4.1.1 The Zhorai Platform

The Zhorai platform is an online web interface with an embedded, teachable conversational agent named Zhorai, as shown in Figure 4-1 [99]. We built the Zhorai platform and curriculum around three of the "Big Ideas" in AI [178]. These ideas allowed us to explore the utility of a conversational interface in teaching key AI concepts to children, which are the basis of current conversational agent technology. The three ideas include:

- Representation and reasoning: Children are expected to understand how Zhorai learns and represents new information. Zhorai generates two different visualizations to show its knowledge representation.

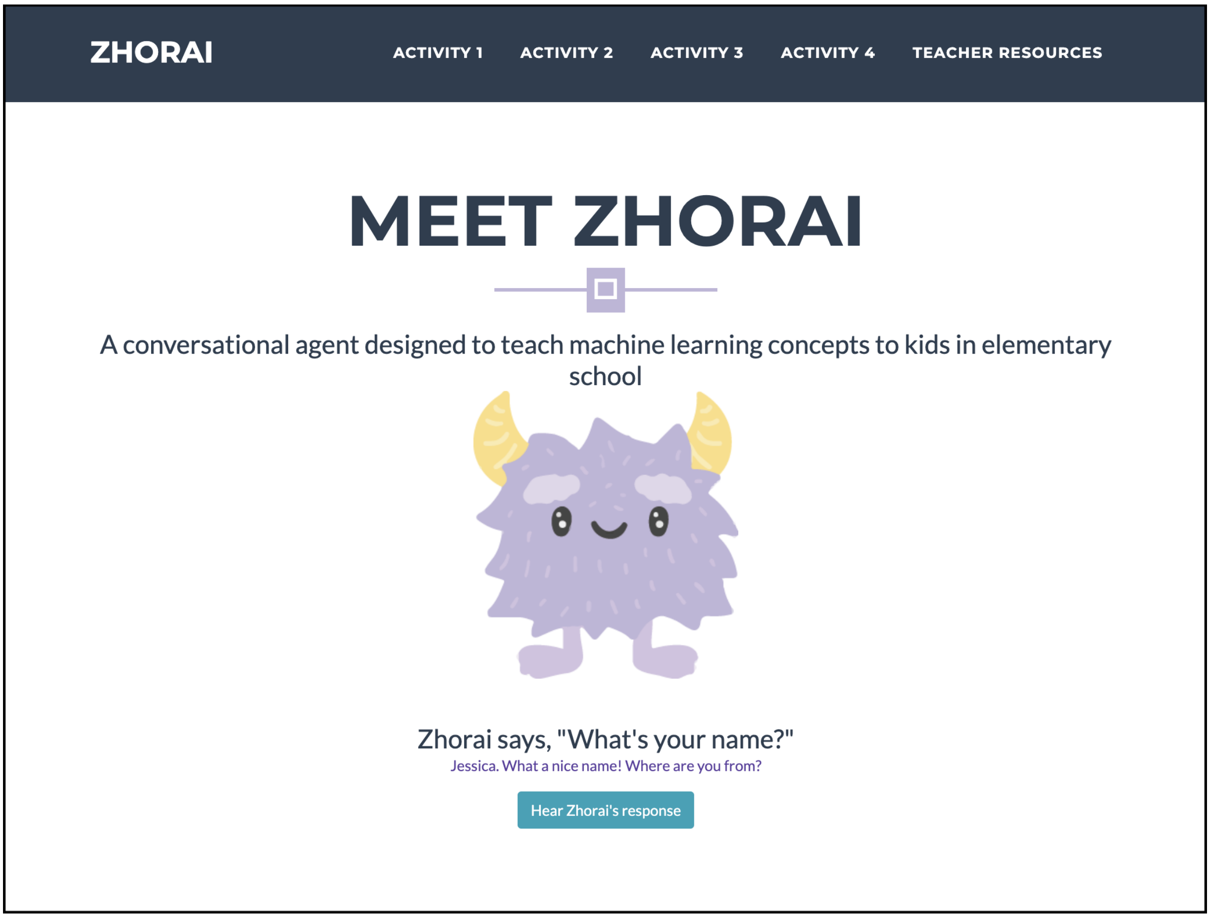


Figure 4-1: The introductory webpage to the Zhorai activity.

- Machine learning: Zhorai also demonstrates the concept of how machines classify concepts. Children witness instances when Zhorai might succeed or fail in its learning, and make attempts to correct the agent.
- Societal impact: The curriculum emphasizes the ethics and societal impact of AI through structured discussion with a facilitator and thinking about implications for bigger-picture contexts.

In the activity, children engage with Zhorai through conversation. Zhorai explains that it is an alien visiting Earth that wants to learn about all of Earth's life (i.e., ecosystems on Earth). The curriculum is focused on Earth's ecosystems because children can describe ecosystems (defined as places where animals live) without much prior knowledge. Additionally, many state science standards discuss the concept of ecosystems in our target grade band of 3-5 [121].

System Design

We designed the Zhorai platform with elementary school students, teachers, and ease-of-use in mind. It can be run using Google Chrome anywhere with internet access. The main components of the system are the (1) **speech synthesizer**, (2) **speech recognizer**, (3) **semantic parser**, (4) **classifier**, and (5) **website visualizer**. Figure 4-2 illustrates the platform's architecture and user flow. Figure 4-3 shows the first module's user interface, in which students teach Zhorai by recording sentences about animals. Further details about how we designed the system can be found in our paper [99].

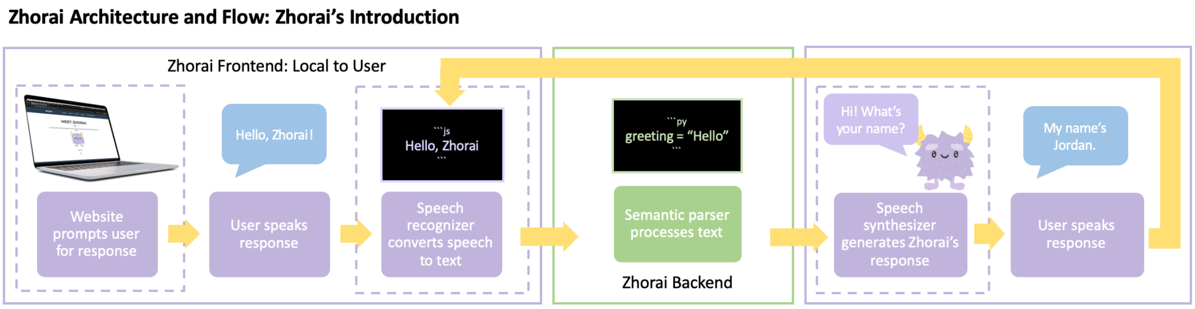


Figure 4-2: A representation of the introduction to Zhorai in terms of architecture and user flow [99].

4.1.2 Zhorai Activity Analysis

The concepts taught in this activity are described throughout the following analysis and outlined in Table 4.3. The Zhorai curriculum is embedded within the Zhorai website, and consists of four modules: "What Does Zhorai Know?", "Teaching Zhorai", "Witnessing Machine Learning", and "AI and Ethics". Typically, the curriculum is completed with a small group of students and a teacher or facilitator. Before entering the first module, children learn to interact with Zhorai through an introduction, in which Zhorai greets and converses with them. In the following sections, I describe how a fictional student, Morgan, might engage with the materials, illustrate how he might learn different conversational agent concepts, and outline the concepts the Zhorai activity teaches.

Activity Introduction

Children first interact with Zhorai by introducing themselves. To illustrate, a conversation between a fictional student, Morgan, and Zhorai follows:

Figure 4-3: An image of the user interface in which students teach Zhorai by recording sentences about different animals. Students click and hold on the mic button to record sentences, which appear below the button. The recorded sentences have red buttons to their right, which allow students to delete them.

Morgan: Hello

[Morgan notices the word he spoke, "Hello", appear below the fluffy, purple-and-yellow alien on the interface. The alien then begins to respond and more text appears on-screen. The alien has a squeaky, high-pitched voice.]

Zhorai: Hi there! I don't remember meeting you before. What's your name?

Morgan: [Mumbles] My name's Morgan[1]

[The text, "More than", appears in the interface.]

Zhorai: [Mis-recognized Morgan's response] Sorry, what was that?

Morgan: Oh! I'm Morgan!

Zhorai: Nice to meet you, Morgan! Where are you from?

Morgan: Canada

[1]Note that all information given to Zhorai is only stored on the user's local computer for privacy reasons.

```
Zhorai: Interesting! I'm from planet Igbruhmmelkin. I've never heard of Canada before.
    Can you tell me more?
[A button appears on-screen, stating "Find out what Zhorai knows about your planet!",
    which takes Morgan to the first module.]
```

As shown in this example, if Zhorai's speech recognition does not work correctly, it will respond with something like, "Oh, pardon?", allowing the child to respond again. Through this relatively simple introduction, children encounter foundational conversational agent concepts. These include:

- **Textual representations** (Conversation Representation): As shown in Figure 4-1, Zhorai's speech output is represented by purple text in the interface. Students will also observe their speech being recognized and represented by text as they interact with Zhorai. This simple representation allows them to visually understand the conversation, and whether Zhorai recognized their speech correctly, like how "Morgan" was misrecognized by Zhorai as "more than" in the example conversation.
- **Turn-taking** (Dialog Management): When interacting with Zhorai, students take turns speaking back-and-forth. Zhorai starts its turn after a student completes a speech recording; Students start their turn when they click the record button. Students can understand their turns as "input" to the system, and Zhorai's turns as "output".
- **Speech synthesis** (Human-Agent Interaction): Students encounter speech synthesis when they hear Zhorai's high-pitched, alien-like voice. Its voice is clearly not human, but generated by a computer. In other activities, like the Alexa Developer Console, students encounter very human-like, computer-synthesized voices. Speech synthesis is an example of computer output in a conversational system.
- **Speech recognition** (Human-Agent Interaction): In the Zhorai activity, when students click and hold the mic button, their speech is recorded as sound waves, which are eventually converted to text. This process is called speech recognition, and is one method for a computer to receive user input in a conversational system. Sometimes, speech recognition systems do not work as expected. For example, if a student says the word, "Zhorai", to Zhorai, it may recognize it as "sore eye" or something similar-sounding, as the word, "Zhorai", is not in its speech recognition vocabulary.

- **Recovery** (Human-Agent Interaction): When Zhorai does not understand students' speech (e.g., when Zhorai recognized the words "more than" instead of "Morgan" in the example conversation), it responds with something like, "Oh, pardon?". This indicates that Zhorai did not understand and would like the user to respond another time, allowing for recovery of the conversation.

This interaction with Zhorai also introduces conversational design practices, including "G1: Visibility/Feedback of System Status" by visualizing recognized speech and speech output as text; "G7: Flexibility and Efficiency of Use" by allowing students to interrupt Zhorai; "G8: Minimalism in Design and Dialogue" through providing implicit confirmations, like restating the country Zhorai heard from the student; and "G10: Providing Help and Documentation" through a scaffolded introduction to interacting with Zhorai. Zhorai's high-pitched, child-like, gender-neutral voice also addresses the design recommendation, DR-U2, which emphasizes developing "diverse, multilingual conversational agent personas that many people can relate to and understand". Table 4.2 summarizes the recommendations the activities addressed.

Module 1: "What Does Zhorai Know?"

The first curriculum module, "What Does Zhorai Know?", introduces children to knowledge representation and reasoning using concept maps. In this module, children ask Zhorai about the five ecosystems that it knows: deserts, oceans, grasslands, rainforests, and tundras. The following description shows how the fictional student, Morgan, might interact with Zhorai:

```
[A prompt appears on-screen below Zhorai stating, "Ask Zhorai about earth's ecosystems
     by saying things like, 'What do you know about the desert?'"]
Facilitator: Let's see what Zhorai knows about different ecosystems! Morgan, could you
     ask Zhorai about the desert?
Morgan: Okay! [Presses record button.] Zhorai, do you know anything about the desert?
Zhorai: I've heard about deserts before! Here's what I know about them.
[Sentences about the desert that "Zhorai heard from other Earthlings" appear, as well
    as a related mind map, as in Figure 4-4.]
Morgan: [Referring to the colored bubbles on the mind map.] Ooo cool! What are the
    colors for?
Facilitator: Can you guess? It has something to do with the sentences that Zhorai was
    told -- or, in AI-speak: the sentences Zhorai was "trained on".
```

Morgan: Hmm... Maybe the blue words are a part of the desert, but the orange ones aren't?

Facilitator: Totally! If you look at the sentences, the orange words are *negated*, or have words like "not" or "don't" in front of them. Like how the word, "water", is negated in the sentence, "Deserts don't have much water".

Morgan: Interesting! So that's how Zhorai knows if the desert *is* or *isn't* like the words in the mind map.

Facilitator: That's how Zhorai represents it, yes! And Zhorai actually figures out whether a word should be orange or blue through the process of *semantic analysis*, which is really just a big term for understanding the meaning of language.

The representation method Zhorai uses is a undirected graph or "mind map", which children can analyze to determine ecosystem attributes. As described in the above example, positive attributes (what an ecosystem has) are visualized as blue circles, and negative attributes (what an ecosystem does not have) are visualized as orange circles, as in Figure 4-4. Children also analyze the corpus that Zhorai is given to form the mind maps, and thus draw connections between natural language sentences and details of the corresponding mind map.

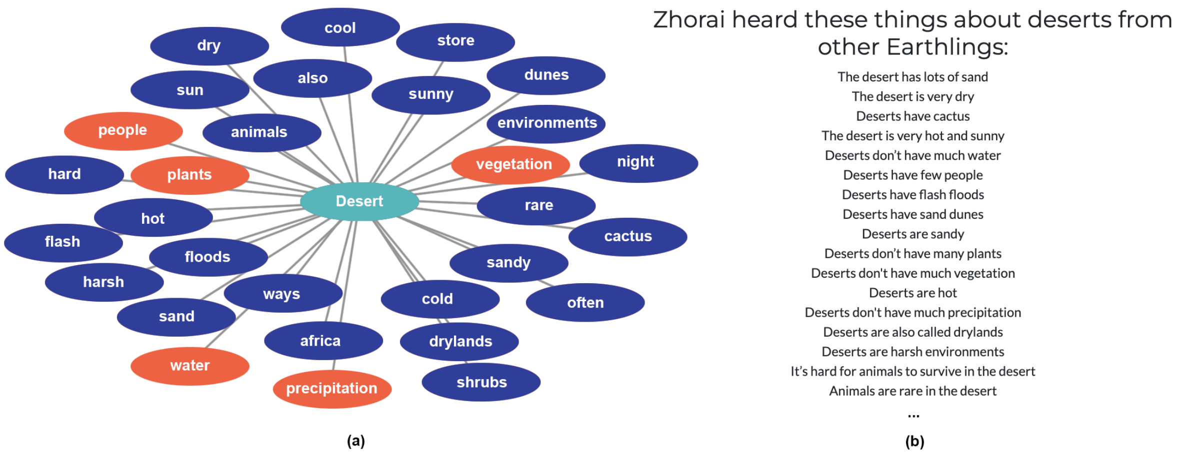


Figure 4-4: The mind map (a) representing Zhorai's understanding of the corpus of sentences (b).

The key concepts developed in this module include:

- **Undirected graphs** (Conversation Representation): Zhorai represents its natural language understanding through a "mind map" or simple undirected graph. Undirected graphs visualize ideas through connecting concepts to each other, without any hierarchy or direction. In the Zhorai activity, Zhorai represents sentences through connecting associated words, like "sunny", "sand" and "water", to a central concept, like the desert. In this representation, words that have a positive association to the central concept, like "sand", appear in blue, whereas words with a negative association, like "water", appear in orange.

- **Semantic Analysis** (Natural Language Understanding): Students encounter semantic analysis when they observe how Zhorai understands sentences with negation. For example, when Zhorai analyzes the sentence, "Deserts don't have much water", it creates an orange (negative) bubble for the word, "water", whereas when it analyzes the sentence, "Oceans are filled with water", it creates a blue (positive) bubble. This serves as an initial introduction to the large field of natural language understanding and semantic analysis.
- **Pre-programmed data** (Data Access and Conversation Context): Students first encounter data access in the Zhorai activity when they ask Zhorai about what it knows. This causes a number of sentences to appear, which were pre-programmed into Zhorai's memory. This pre-programmed data is opposed to the data students teach Zhorai in later modules. Zhorai initially knows about different ecosystems, like the desert, tundra and ocean, but does not know anything about the animals who live there. This gives students the opportunity to provide Zhorai with additional information about animals, which has not been pre-programmed into its memory.

In this module, Zhorai provides a prompt ("Ask Zhorai about earth's ecosystems by saying things like, 'What do you know about the desert?' ") for Morgan. This addresses the design recommendations, "G6: Recognition Rather than Recall" and "G10: Providing Help and Documentation", as shown in Table 4.2.

Module 2: "Teaching Zhorai"

In the next module, Module 2, "Teaching Zhorai", children are tasked with providing Zhorai with data about three animals of Zhorai's choosing, as shown in Figure 4-3. An example interaction between Zhorai and Morgan follows.

```
[Morgan enters the second module.]
Facilitator: Now let's teach Zhorai about a few animals, so that it can guess which
    ecosystems they live in! It looks like Zhorai wants us to teach it about camels
    right now. Do you want to say a few things about camels to Zhorai, Morgan?
Morgan: Okay... but how do I say it?
```

```
Facilitator: You can say it any way you want, but just make sure you don't say the
    word "desert", or that will give away the ecosystem to Zhorai. When you're ready,
    you can press the record button to start.
[Morgan nods, presses the record button, and starts talking to Zhorai.]
Morgan: Camels live where it's really hot and dry.
[Morgan's sentence appears on-screen.]
Morgan: Camels store water in their backs.
Morgan: [To facilitator] I don't know much more about camels... Can I look them up?
Facilitator: Sure, let's find some more fun facts together!
[Morgan and the facilitator find some additional facts online. They realize that
    camels don't actually store water in their backs, but they store fat, so they
    delete the sentence about water.]
Morgan: Camels store fat on the top of their backs so that they don't get too hot.
Morgan: Camels keep sand out of their eyes with extra eyelids!
Morgan: Camels can eat thorny plants because they have very thick lips.
Morgan: Camels can carry 500 pounds on their backs!
Morgan: [To facilitator] Okay, I'm done!
Facilitator: Alright, you can click the button to let Zhorai know.
[Morgan clicks the button stating, "What is Zhorai thinking?".]
Zhorai: Camels sound fascinating! Now I want to visit earth and all of it's life! I'll
     show you what I understand after I think for a little while.
[After Zhorai analyzes the data Morgan entered, a mind map of its understanding
    appears on the screen, and another prompt about teaching Zhorai about "polar
    bears" appears. Morgan teaches Zhorai about polar bears and a few other animals,
    and then moves on to the next module.]
```

As illustrated by this example, Zhorai functions as a less-knowledgeable, teachable AI system. Throughout the process of teaching Zhorai, children can see what Zhorai heard and whether it misheard certain words, and can modify sentences as necessary, as shown in Figure 4-5 (b).

The key concepts developed in this module include:

- **User-defined data** (Data Access and Conversation Context): Students can provide Zhorai with data by telling it sentences about animals. The sentences appear in the user interface, as in Figure 4-3, and cause Zhorai to create an undirected graph, similar to the one in Figure 4-4. Students can observe how when they modify sentences, Zhorai's understanding shown in the graph also changes. This ties into the concept of machine learning, as machine learning models can either be pre-

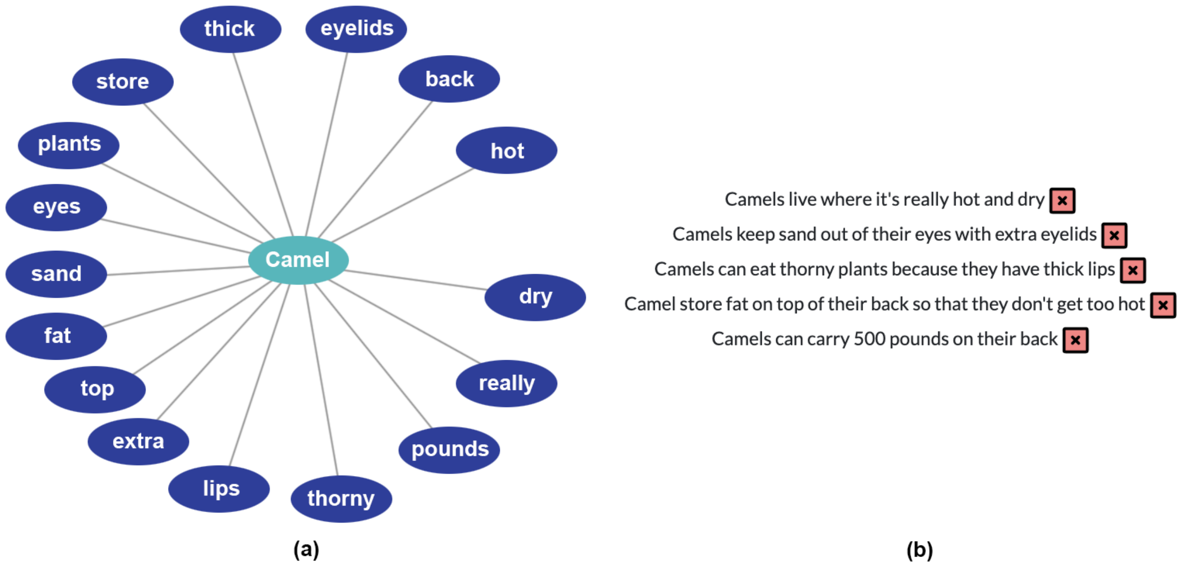


Figure 4-5: A mind map (a) generated according to the sentences (b) given to Zhorai by Morgan in Module 2.

trained (e.g., with pre-programmed ecosystem data) or trained in real-time (e.g., with the animal data students input).

- **Recovery** (Human-Agent Interaction): In addition to the out-of-vocabulary recovery encountered in Zhorai's introductory activity, students will likely encounter recovery when telling Zhorai sentences about animals. For example, Zhorai might misrecognize "A group of polar bears is a 'sleuth' " as "A group of polar bears is aloof". Students can identify misrecognized sentences with the textual representation in the interface. If a sentence was misrecognized, they can recover the interaction by deleting the sentence (by pressing the red 'X' buttons shown in Figure 4-3) and re-recording it.

This activity also addresses the conversational design practices of "G3: User Control and Freedom" and "G9: Allowing Users to Recognize and Recover from Errors" by allowing students to identify whether Zhorai recognized the sentences they told it correctly, delete them if misrecognized, and update them accordingly.

Module 3: "Witnessing Machine Learning"

In Module 3, "Witnessing Machine Learning", children observe Zhorai's learning and reasoning process. They ask Zhorai to guess which ecosystem it thinks the animals they previously taught it about are from. An example of this interaction follows:

```
[A prompt, stating "Ask Zhorai to guess where animals live", appears on screen.]
Morgan: Hey Zhorai! Where do you think camels live?
Zhorai: Oh yes! Let me think about camels for a second.
[After computing similarity scores between the sentences about camels and the
    ecosystems, a histogram appears, as shown in Figure 4-6.]
Zhorai: Based on what I know about Earth, I would guess camels live in grasslands.
Morgan: Ha ha! Zhorai doesn't know that camels live in the desert!
Facilitator: Aw too bad! We must not have taught Zhorai well enough. From the
    histogram, though, it looks like the desert would have been its second guess.
    Let's go back to our sentences and figure out why Zhorai guessed grasslands
    instead of the desert.
[They return to the page where Morgan trained Zhorai, as in Figure 4-5.]
Facilitator: Do you see anything in the mind map that might make Zhorai think camels
    are from the grasslands?
Morgan: Hm... Well, the word, "plants", is pretty similar to "grass", so maybe that's
    why?
Facilitator: It definitely could be! And if we go back to the mind map about the
    grasslands... Look at that! There are lots of similar words: "grass", "shrubs"...
    and even the word, "warm", which is similar to "hot" in our camel mind map.
Morgan: Okay, so maybe we can tell Zhorai things about camels that are in the desert
    mind map... Like the words, "dunes", "harsh", "dry", and "sunny".
Facilitator: Let's do it!
[Morgan goes back to Module 2 to record a sentence.]
Morgan: Camels live on harsh, dry dunes where it's sunny all the time!
[Zhorai's mind map updates to include Morgan's additional desert-related words. They
    return to Module 3 and ask Zhorai where it thinks camels live. Another histogram
    appears, as shown in Figure 4-7.]
Zhorai: I would guess camels live in deserts from what I know.
Morgan: Yes! We did it!
Facilitator: We totally did! We trained Zhorai to understand the connection between
    camels and deserts, just by giving it similar sentences. Awesome job, Morgan!
```

In this example, Zhorai creates two histograms with its ecosystem-guesses (as shown in Figures 4-6 and 4-7) by comparing the words representing the animal and the words representing the ecosystems, and computing word similarity scores for each ecosystem. For each animal, Zhorai chooses the ecosystem that has the highest similarity score among the five ecosystems for its final guess. It is important for children to understand when and why Zhorai may guess incorrectly by drawing connections between similarities within

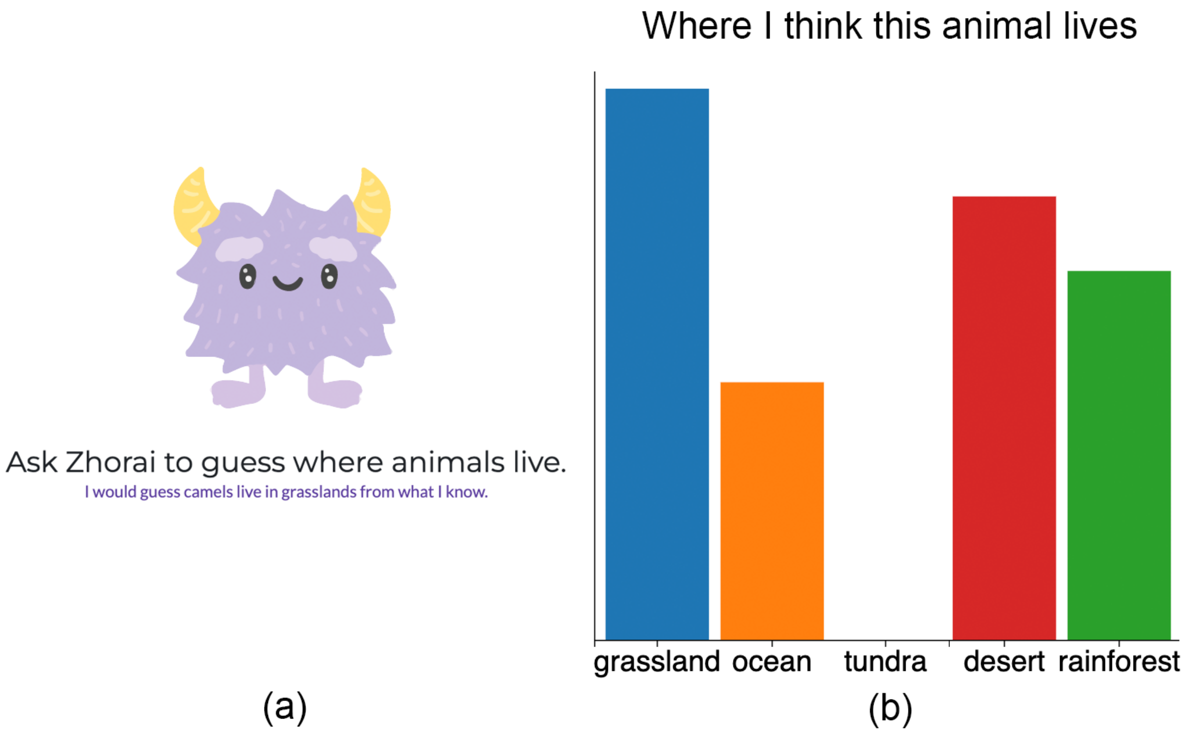


Figure 4-6: The initial histogram (b), in which Zhorai guesses camels live in grasslands (a). This graph is generated through computing similarity scores between the descriptions of ecosystems and camels, as described in our paper [99].

the mind maps of Module 2 and the scores of Module 3, and how to manipulate this connection.

The key concepts developed in this module include:

- **Machine Learning** (Natural Language Understanding): Students encounter aspects of machine learning throughout the Zhorai activity, especially when they observe Zhorai classify animals into ecosystems. Based on students' sentences about animals, Zhorai matches patterns between the pre-programmed sentences about ecosystems, like "The desert is very hot and sunny", and user-defined sentences about animals, like "Camels can endure hot temperatures over 50 degrees Celcius". If Zhorai incorrectly classifies the animal, students can modify the training data and observe how the model "learns" or changes its guesses.
- **Similarity scores** (Natural Language Understanding): Zhorai computes similarity scores between animals and each ecosystem, as shown on the histogram in Figure 4-6, by comparing the words in each animal sentence with those in each ecosystem sentence. Zhorai guesses the animal is from the ecosystem with the highest computed similarity score (i.e., the grasslands in Figure 4-6). To attempt to cause

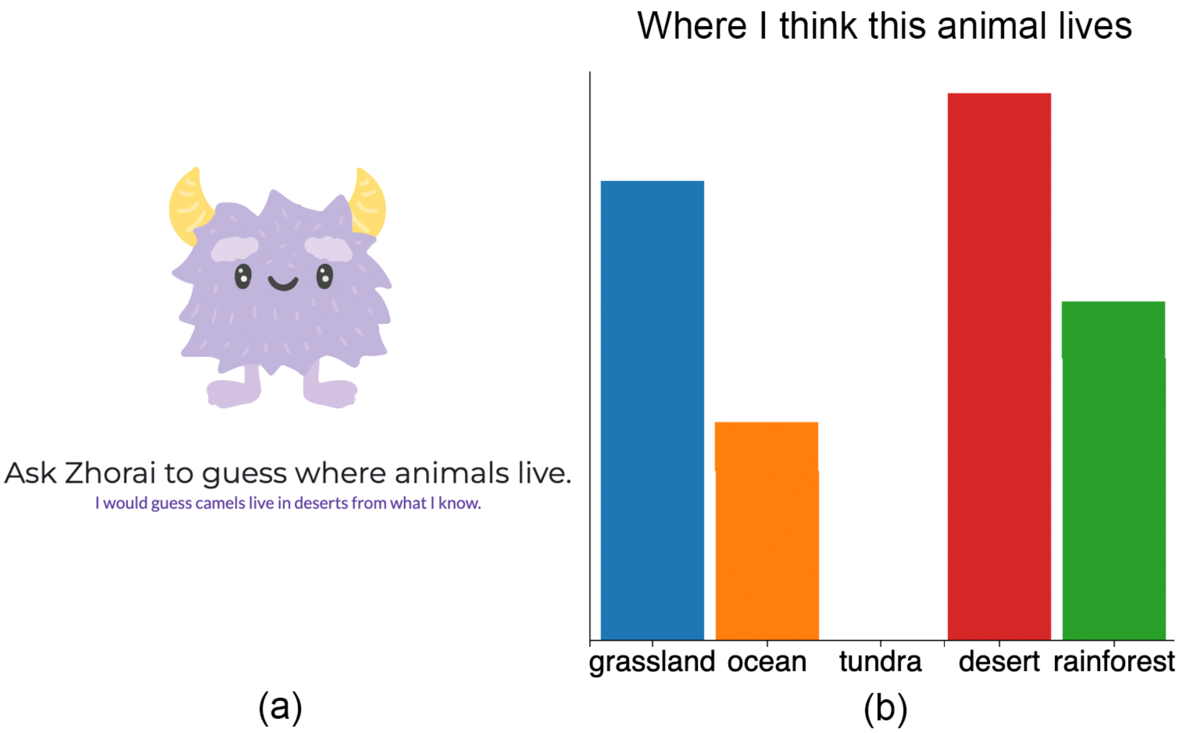


Figure 4-7: The second histogram (b), in which Zhorai guesses camels live in deserts (a). This graph is generated through computing similarity scores between the descriptions of ecosystems and camels, as described in our paper [99].

Zhorai to guess the correct ecosystem, students can modify their animal sentences to be increasingly similar to the pre-programmed ecosystem sentences.

- **Histograms** (Conversation Representation): The bar charts or histograms in Figures 4-6 and 4-7 are representations of similarity scores. This type of representation allows students to identify which ecosystem has the highest similarity score, or in other words, which ecosystem Zhorai will guess the selected animal lives in. In natural language understanding modeling, histograms can represent many different concepts, including word counts and other distributions.

Module 4: "AI and Ethics"

In the last module, "AI and Ethics", facilitators lead a discussion about how conversational agents and agents that learn from data are used in society with positive and negative consequences. These discussion questions are scaffolded in the Teacher Resources section on the website. Children reflect on instances when Zhorai made mistakes, and are asked questions like, "Would Zhorai know whether what we teach it is correct or not?" and "How would you feel if Zhorai learned something untrue about you?", as shown in Figure 4-8.

The goal of this module is to empower children with the tools to design AI with ethics in mind. We also probe children about what the societal impact of mistakes made by AI can be, and how we can mitigate harm [138, 99]. In Appendix E, there are additional educational materials to help teachers facilitate such discussions.

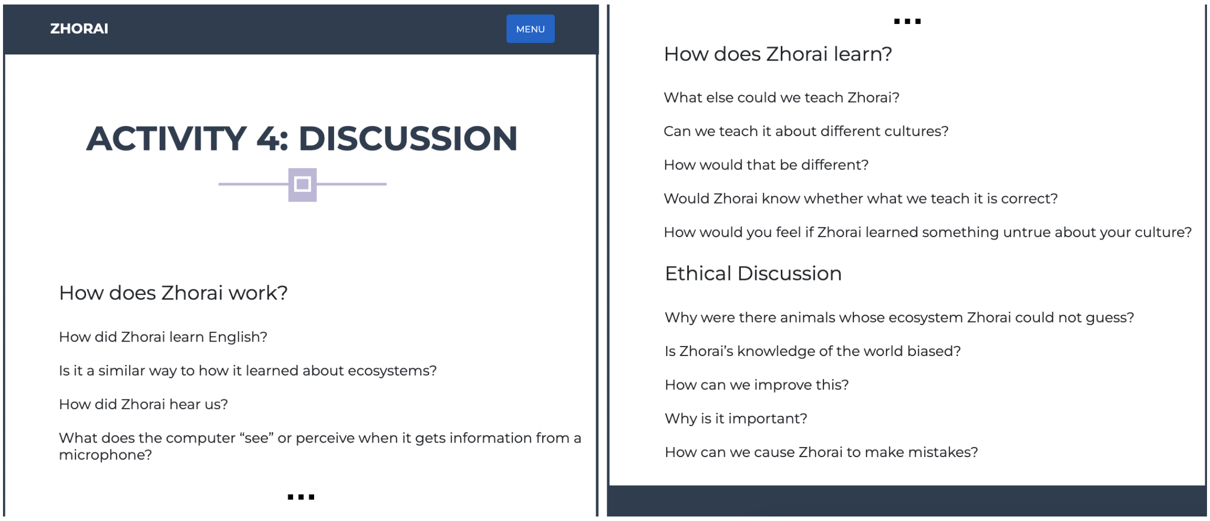


Figure 4-8: The discussion questions provided in Module 4.

This module focuses on the key concept of *societal impact and ethics*:

- **Societal impact and ethics** (Human-Agent Interaction): Conversational agents are becoming commonplace and affect many people's current lives. In the Zhorai activity, students discuss the implications of this, including how Zhorai and other conversational systems can be biased and make mistakes, how different applications of this technology may not be appropriate (e.g., classifying humans into certain categories), and why it is important for developers and users alike to understand the technology.

4.1.3 Summary of the Zhorai Analysis

Through the Zhorai activity, students can learn the conversational agent concepts outlined in Table 4.3. They can also learn conversational agent design best practices through interacting with Zhorai and observing how it works. These best practices are outlined in Table 4.2.

Table 4.3: List of conversational agent concepts the Zhorai activity teaches, organized by concept category.

Category	**Concept**
Natural Language Understanding	Machine learning Similarity scores Semantic analysis
Conversation Representation	Textual representations Undirected graphs Histograms
Dialog Management	Turn-taking
Data Access and Conversation Context	Pre-programmed data User-defined data
Human-Agent Interaction	Speech synthesis Speech recognition Recovery Societal impact and ethics

4.2 Activity 2: ConvoBlocks

As described in Section 3.1, ConvoBlocks (aka the MIT App Inventor Conversational Agent Platform) empowers nearly anyone to code conversational agent programs for Alexa. Students develop their agents using block-based, visual programming, and can create complex conversations associated with mobile app functionality. For example, the fictional student, "Sheila", from Section 3.1, creates an agent which reads and answers questions about a storybook. She programmed the storybook to appear on an app she developed with MIT App Inventor, and her agent can acknowledge what is on this app. In this section, I provide a brief overview of the system, which is described in more detail in Chapter 3, and discuss the ConvoBlocks curriculum from our paper [191]. In this paper [191], as well as two others [185, 193], we describe studies in which students learn AI and conversational AI concepts by developing agents using ConvoBlocks. Additional teaching resources for the ConvoBlocks activity can be found in Appendix F and the ConvoBlocks code repository can be found on GitHub [192].

4.2.1 The ConvoBlocks Platform

ConvoBlocks is a block-based programming tool for creating conversational agents. The code blocks range from "say" blocks, which cause Alexa to say user-defined phrases, to "send to CloudDB" blocks, which send user-defined information to the App Inventor cloud [94], to typical control functions, like "if", "while" and "break" blocks. Figure 4-9 shows many of these blocks. This block-based coding interface empowers students—typically around 11 to 18 years old—to create their own conversational agents, which can be deployed on any Alexa-enabled device, as well as within the MIT App Inventor interface. Figure 4-10 shows a conversation with an example agent in this interface. Students can have audible or text-based conversations with their agents.

More information about our design goals, development process and implementation of ConvoBlocks interface can be found in Section 3.1 and my related publications [185, 191, 193, 186].

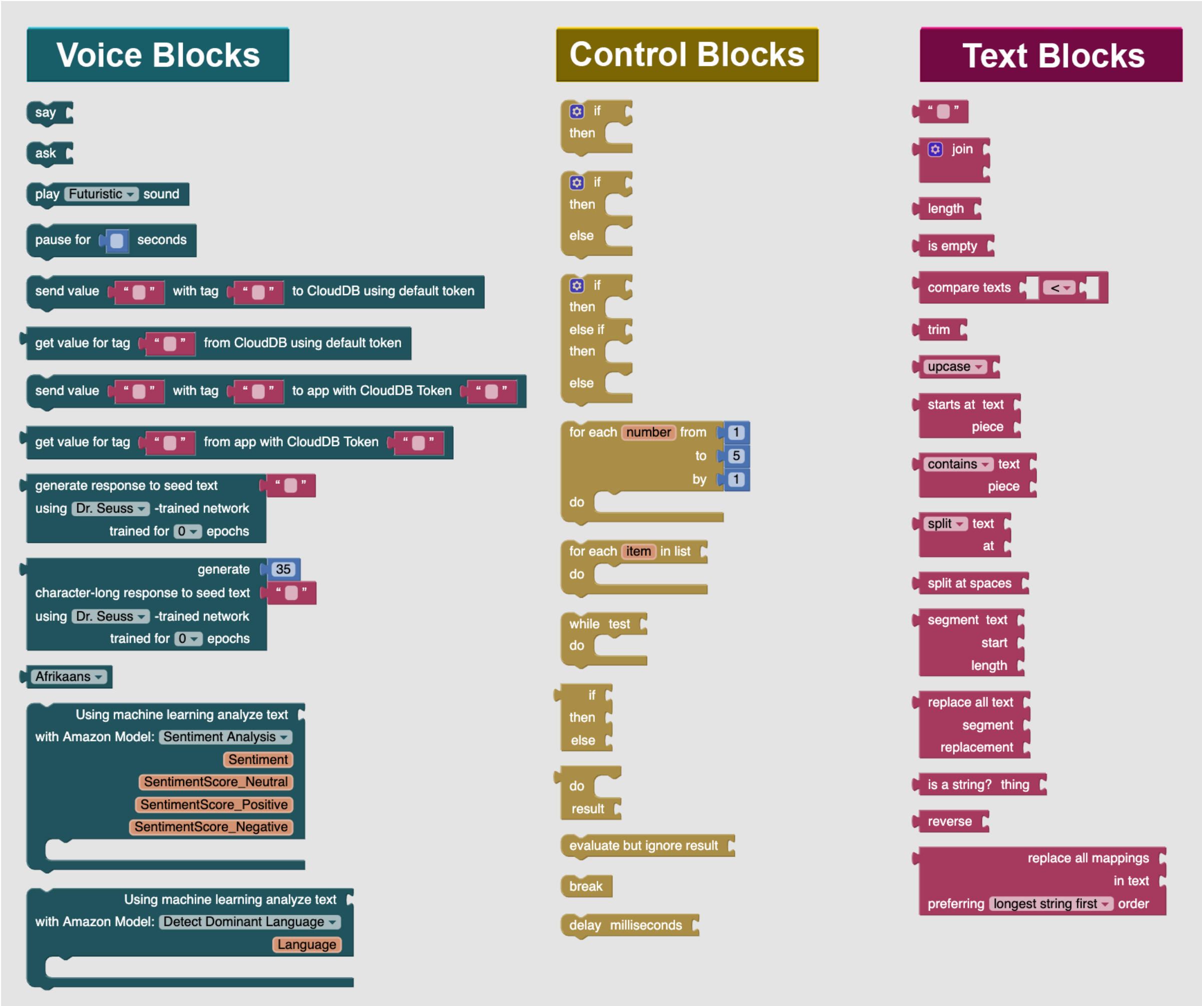


Figure 4-9: A number of the available blocks users arrange to create conversational agents with ConvoBlocks. The "Voice" blocks cause Alexa to complete an action, like saying a phrase; the "Control" blocks change the flow of the program, like only completing an action *if* something is true; and the "Text" blocks define textual data, like joining together two strings.

4.2.2 ConvoBlocks Activity Analysis

As described in our paper [191], the ConvoBlocks activity consists of an introduction to the MIT App Inventor environment (where ConvoBlocks resides), block-based programming and a simple rule-based conversational agent; interactive lectures and group discussions on the "Big 5 AI Ideas" [178], conversational AI, and AI ethics; and conversational agent development tutorials and final projects. The following sections describe how the curriculum teaches fundamental conversational agent concepts.

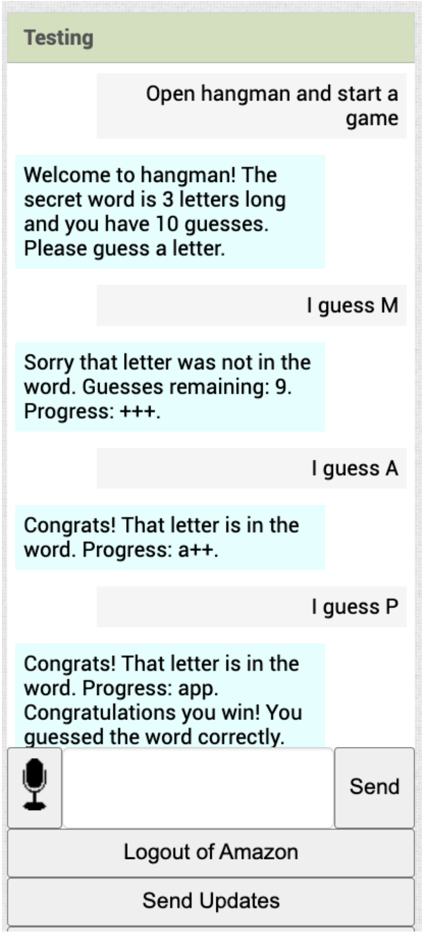


Figure 4-10: An example conversation in ConvoBlocks with a "hangman game" agent. Users can develop this agent by combining Voice, Control, Text, Lists, Procedures, Variables and Math blocks; test it in the interface itself; and deploy it to any of their Alexa-enabled devices.

Activity Introduction

As an introduction to the ConvoBlocks activity, students complete mobile app-building tutorials in MIT App Inventor, and develop familiarity with programming concepts relevant to conversational AI, such as variables, control statements, and events. The apps students develop include a counter for button presses and a rule-based conversational agent. The agent can recognize the words, "hi" and "goodbye", and respond with programmed-phrases, as shown in Figure 4-11. This activity introduces students to the difficulty and repetitiveness of developing rule-based agents (as they cannot generalize over phrases, but rather only understand specific phrases), and provides a segue into developing machine learning-based agents.

The main conversational agent concepts students encounter in the introduction include:

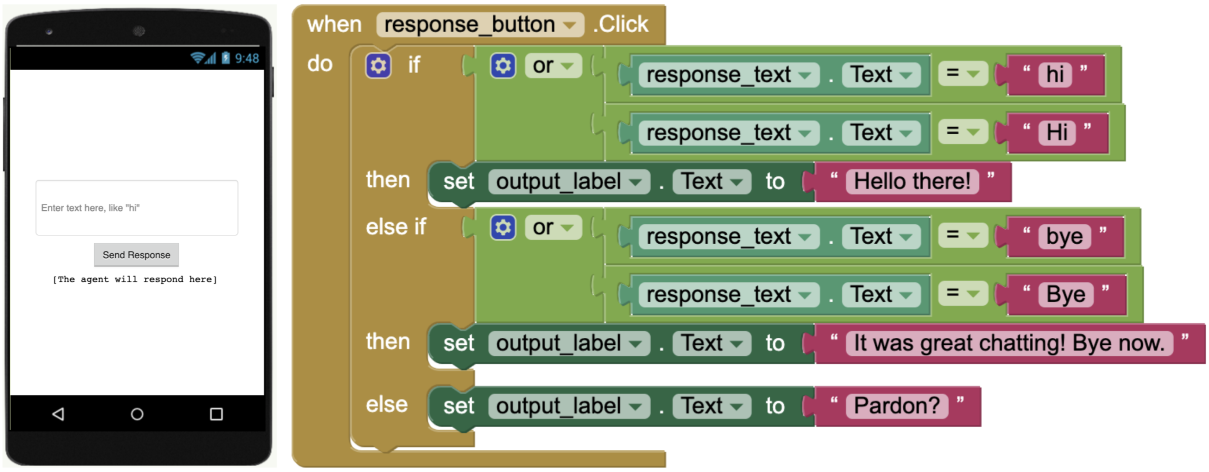


Figure 4-11: The rule-based agent app students develop in the ConvoBlocks introduction. This app has an event handler for when an end user clicks a button. When this occurs, the program compares the user-provided text to "hi", "Hi", "bye" and "Bye", and provides an appropriate response. Later on in the curriculum, students learn to develop agents that use machine learning to generalize over phrases, so that they do not have to enter every possible phrase similar to "hi" or "bye".

- **Events** (Dialog Management): Students define events and responses (or "event handlers") to manage agents' dialog. For example, a student might define an event to occur when an end user states, "Hi", and define the response from the agent to be "Hello there!". Students can define many events and responses, which—when combined—define larger agent conversations.
- **Conditions** (Dialog Management): With MIT App Inventor and ConvoBlocks, students can define whether or not particular pieces of dialog or events occur by adding conditions to their event handlers. For instance, if an event is triggered when an end user states something about their favorite number, the student can create a condition that causes the agent to say, "Wow, that's a large number!", for anything greater than or equal to 100 and, "That's a pretty small number compared to 100!", for anything below. This is shown in Figure 4-12 with an "if-statement" condition.
- **Event-driven program representations** (Conversation Representation): In MIT App Inventor and ConvoBlocks, conversations are represented through event-driven block-based programming. This means each each potential trigger event is represented as a block of code, as shown in Figure 4-13, in which there is a block of code for the "hi" event and the "bye" event. With large or complex conversations, this representation can get convoluted and difficult to follow. In the Dialogflow activity,

developers use another method of conversation representation, which is useful for developing especially complex agents.

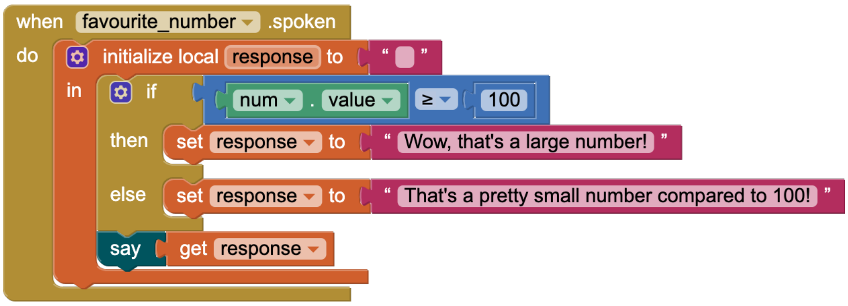


Figure 4-12: An event handler for when someone says their favorite number. The handler contains an "if-statement" condition such that when the number is greater than or equal to 100 the agent responds differently than when it is smaller.

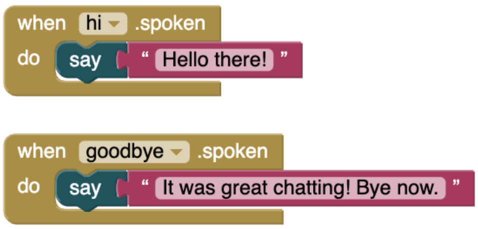


Figure 4-13: Two event handlers for the "hi" and "bye" events students may define with ConvoBlocks. This program utilizes machine learning to recognize intents, rather than a rule-based approach.

Lectures and Group Discussions

In this part of the ConvoBlocks activity, students listen to presentations by facilitators and engage in group discussions. The presentations discuss the "Big 5 AI Ideas" [178], conversational AI, and AI ethics. For example, students discuss whether or not automatic captioning, medical bots and text-based image search are conversational AI systems and why. They also learn about how machine learning, large language models and transfer learning can enable them to create agents that generalize across phrases to understand specific "intents" rather than just specific character-by-character words. Finally, they discuss the implications of conversational AI, including gender and racial bias in speech recognition, the ethics of extremely human-like agents, and deepfakes [172, 19].

The main conversational agent concepts students encounter in this part of the ConvoBlocks activity include:

- **Training** (Natural Language Understanding): In the ConvoBlocks activity, students learn about how machine-learning-based conversational agents can be trained to recognize phrases that are *similar* but not necessarily exactly the same as the data provided to the agent. For instance, if a student trained an agent on phrases such as, "Hi", "Hello", "Yo", and so on, the system might also recognize phrases with the same meaning (or "intent") as the training data, like "Howdy", for instance. In the ConvoBlocks activity, facilitators first introduce this concept in a lecture, and students then train their agents and observe this concept in practice.
- **Transfer learning** (Natural Language Understanding): As students learn about the concept of training, they also learn that the training they are performing with their agents is a type of transfer learning. Since the agents they create are built on top of a large pre-trained model (a "large language model") Amazon created, by training their agents with a small amount of additional data, they can enable their agents to discern between meanings (or "intents") of various kinds. For instance, students may train their agents to recognize "hi" and "bye" intents by completing transfer learning training with only 15 to 20 example phrases. This is possible because of the large amount of training Amazon previously completed with the model.
- **Large language models** (Natural Language Understanding): In the ConvoBlocks activity, students learn about how they can quickly train agents to understand different intents due to large amounts of previous training on machine learning models. In this case, because the models are language-based, they are called "large language models". These large language models cluster words (or phrases) in high dimensional space through training on enormous amounts of data [46, 32]. This process often causes words or phrases with similar meanings (e.g., "I am a computer scientist" and "I'm a software engineer") to be clustered together. This also allows developers to perform small amounts of additional training to enable agents to recognize and classify desired intents of phrases, like how students can train agents to recognize greeting or farewell intents with only a small amount of example phrases.
- **Intents** (Natural Language Understanding): In the ConvoBlocks activity, students learn about how they can train agents to recognize phrases with similar meanings, or "intents". For example, both of the words, "hi" and "hello", have the intent to greet

someone, and by providing an agent with a number of similar words or phrases, students can cause agents to recognize this intent. This forms the basis for how the students' agents understand natural language and engage in conversation. For example, students might train an agent to recognize a greeting intent, including training phrases like, "Yo" and "Hello", and then program the agent to respond with another greeting, like "Hi, friend!", which begins a conversation.

- **Societal impact and ethics** (Human-Agent Interaction): As in the Zhorai activity, the ConvoBlocks activity also includes societal impact and ethics curriculum. It encourages students to think deeply about the implications of having conversational agents in the world, as well as how they might create positive, helpful agents. Some of the topics students encounter include, gender and racial bias in speech recognition, the ethics of extremely human-like agents, and deepfakes.
- **Speech synthesis** (Human-Agent Interaction): As in the Zhorai activity, students encounter agents with computer-synthesized voices. In this case, they encounter the Alexa voice, which defaults to a female voice and is quite realistic-sounding. In the curriculum, students discuss whether it would be ethical for a company to call someone over the phone using a conversational agent without telling the person they are speaking with an agent and not another human. They also may discuss whether having assistant agents, like Alexa and Google Home, default to female voices is ethical or whether it perpetuates undesirable stereotypes. Students discuss how it is important to consider the implications of how speech synthesis systems are developed and deployed.
- **Speech recognition** (Human-Agent Interaction): As in the Zhorai activity, students observe agents recognizing their speech. Often students notice that the agent recognizes one person's voice well, but not another person's voice. In groups, they discuss how this is due to the training of the system. For example, historically, developers have trained speech recognition systems on more male voices than female voices. Thus, these systems tend to be better at recognizing males than females [172]. Additionally, speech recognition systems may not have been trained on voices with various accents, and thus may not recognize people with these accents. Students discuss how it is important to ensure diversity in training data.

Students also learn about the conversational agent design recommendation, "G3: User

Control and Freedom", through this part of the activity. To increase end user freedom, agent developers can ensure their agents recognize many different ways of phrasing the same intent. For example, by providing more training examples of greetings (e.g., "Hi", "Hi there", "Hello", "Yo", "Howdy", etc.), agents will be better able to generalize over different greeting phrases, increasing end users' freedom. Increasing training data also generally allows agents to better correctly classify intents. Thus, students also learn about, "G5: Preventing User Errors", through better intent classification. In Section 4.3, the Convo activity expands on these concepts in its discussion of constrained vs. unconstrained natural language.

Tutorials and Final Projects

In this part of the activity, students develop conversational agents with step-by-step tutorials, and on their own with final projects. The first tutorial involves training an agent to recognize a greeting intent. This contrasts the rule-based agent tutorial from the introduction, as it enables the agent to generalize over the greeting intent, rather than only recognizing pre-programmed, specific phrases. This provides students with practical experience with the concepts of training, transfer learning, large language models, and intents taught in the lectures and group discussions.

The second tutorial introduces the concept of "slots" or "entities", which are important pieces of information agents can extract from end users' phrases. For example, if an end user said, "I'd like a pineapple pizza delivered on March 24th at 6 PM", an agent might extract a "pizza type" entity (pineapple), a date entity (March 24th) and a time entity (6 PM). This allows the agent to store information about the conversation and respond contextually.

The final tutorial focuses on teaching students how agents can communicate with other devices. In this case, students develop a mobile app with an input text-box, and an agent that communicates with the app to read the text aloud. The app and agent communicate through CloudDB, a cloud database created for MIT App Inventor [94]. This introduces the concept of storage location to students, and empowers them to create more complex agents that end users can interact with multimodally (e.g., with mobile apps' touch screens).

Finally, students create their own agents as final projects, and are encouraged to consider how they can create socially useful agents, as well as utilize the unique capabilities of mobile devices and conversational agents, including those shown in Figure 4-15. Figure 4-14 shows examples of student final projects. Students can test their agents using the testing interface in ConvoBlocks, which can be used in text-based or voice-based mode. If they have Alexa-enabled devices, like Alexa apps or Alexa smart speakers, they can also deploy their agents to these devices.

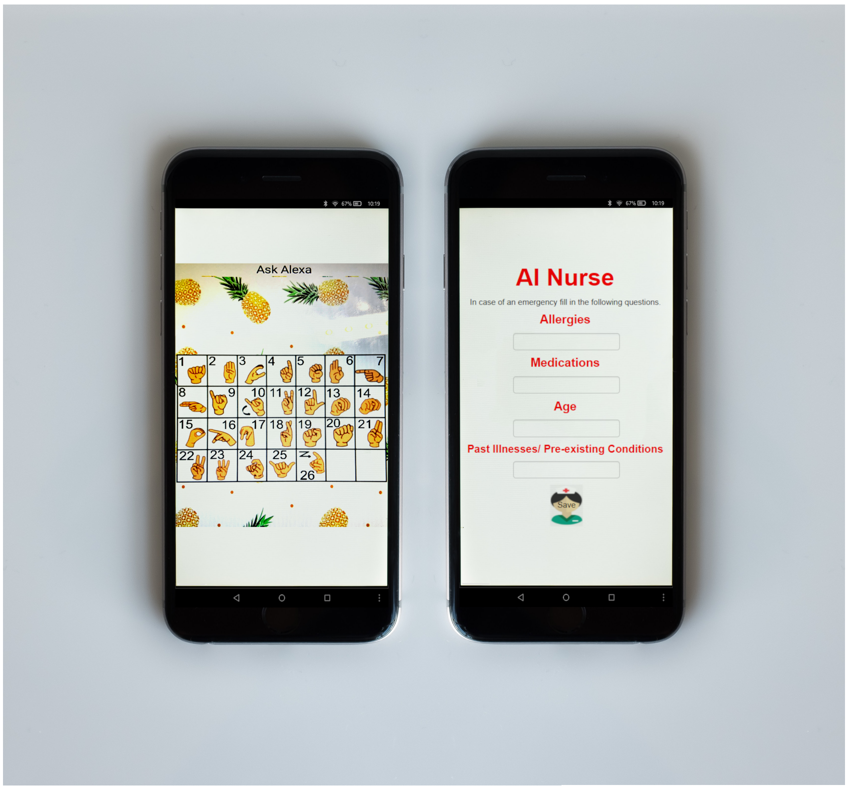


Figure 4-14: Screens from apps for students' final projects. Each app communicated with Alexa-based agents students had designed: one of which helped users learn to sign, and the other, diagnose illnesses.

The main conversational agent concepts students encounter in this part of the activity include:

- **Entities** (Natural Language Understanding): In the ConvoBlocks activity, students learn about entities through creating a custom calculator agent. With this agent, end users provide numbers and the agent uses these numbers to perform some kind of calculation. For instance, an end user might say something like, "What's 11 times 484?". To calculate the answer, the agent needs to extract the two number entities (11 and 484), store them, and multiply them. Other standard entities include dates, times, names, locations, and organizations. Agents can extract and store entities like these and use them later in the conversation.

- **Entity-filling** (Dialog Management): At times, end users do not provide all the information necessary for an agent to complete its task. For instance, if an end user said, "Can you multiply something for me?", without telling the agent which numbers to multiply, the agent would not be able to extract the required number entities to complete a multiplication. With ConvoBlocks, students could program the agent to ask follow up questions like, "What numbers would you like to multiply?". This would allow the agent to manage the conversation until it has extracted the required information to continue the dialog.
- **Contextual data** (Data Access and Conversation Context): In the ConvoBlocks activity, students learn about entity-extraction. The purpose of entity-extraction is to gather information about the current conversational context. By storing entities, like a person's name, agents can engage more naturally with humans, reuse this information later (e.g., by restating the person's name), and show their understanding of the context. Other methods of contextual data access or storage may involve gathering data about current events or the current weather from internet sources, for example.
- **Testing** (Natural Language Understanding): Students encounter testing when using the "testing interface" in ConvoBlocks, as shown in Figure 4-10. The testing interface allows students to identify whether agents recognize particular phrases when end users type or speak them. For instance, students can try entering in the word "Howdy" to see if their greeting-agent recognizes it. If not, students can retrain their agent with additional phrases, and test its recognition again. This pattern of training, testing, retraining and retesting is important to ensure robust conversational agent systems before deploying them to the world.
- **Agent modularization** (Data Access and Conversation Context): When developing with ConvoBlocks, students must enter "invocation names" for their agents. Invocation names are unique identifiers, which allow Alexa systems to identify the particular agent (or "Alexa Skill") end users are trying to access. Different agents have different abilities; for instance, one agent may know how to respond to the question, "What's App Inventor's mascot?", whereas another agent may not. Furthermore, different agents may react differently to the same phrase. For instance,

when asked, “Where are you located?”, one agent might respond with, “Our offices are located at 32 Vassar Street in Cambridge”, whereas another agent might respond with, “Second star to the right, and straight on till morning”. In this way, agents are modularized, and cannot access information or abilities of other agents, unless that data is intentionally shared (e.g., through cloud computing).

- **Device access** (Data Access and Conversation Context): When creating agents with ConvoBlocks, students must click a button to “Send updates to Amazon”. When this occurs, ConvoBlocks sends the programmed information to Amazon Web Services, which enables various devices to access the agent. This does not allow any device to access the agent (e.g., Siri, the Google Assistant, and the developer’s friend’s Alexa app will not be able to access it); however, all Alexa-enabled devices associated with the developer’s Amazon account will be able to access the agent. Through this experience, students learn that agents must be deployed to particular devices or locations (e.g., an Alexa device, a Google home device, the world wide web) in order for end users to access them.
- **Cloud computing** (Data Access and Conversation Context): Students encounter the idea of cloud computing in two different ways during the ConvoBlocks activity. The first is when students send their agents to the Amazon Web Services cloud, which allows them to speak with their agent in the ConvoBlocks interface or on any Alexa-enabled device. They also encounter this idea when they use CloudDB to store information. The mobile apps and agents they created with ConvoBlocks can access this information from where they stored it in the cloud. Students learn how the cloud enables cross-device communication and data access.
- **Text-based interaction** (Human-Agent Interaction): With the ConvoBlocks’ testing interface, as shown in Figure 4-10, students can interact textually (or through speaking) with their agents. Students may notice how textual input tends to have higher accuracy than voice input, as voice input must go through the speech recognition process. The concept of text-based vs. voice-based interaction is explored further in the Convo activity.
- **Voice-based interaction** (Human-Agent Interaction): The typical interaction mode with ConvoBlocks-developed agents is through speaking with Alexa. Students discuss the pros and cons of voice-based interaction, including how speech is

very efficient and natural, but can be inaccurate. The concept of voice-based vs. text-based interaction is explored further in the Convo activity.

- **Multimodal interaction** (Human-Agent Interaction): Students develop mobile apps that can interact with their voice-based agents during the ConvoBlocks activity. They discuss how multimodal interactions like these can be helpful for visibility of the system status (e.g., by showing the conversation history), for including those who have hearing difficulties (e.g., by enabling visual output), and for making a more engaging experience (e.g., by showing images on-screen).

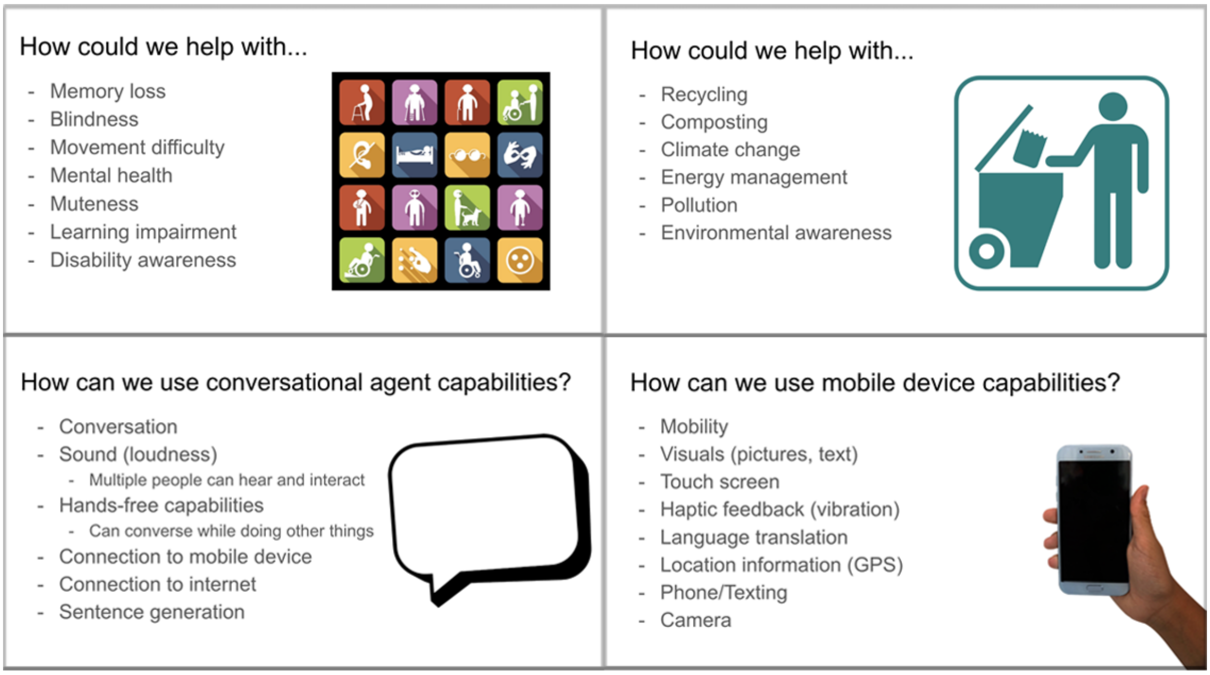


Figure 4-15: Slides from the final project presentation in the ConvoBlocks activity, encouraging students to consider creating socially useful agents, and use the unique capabilities of mobile devices and agents.

By enabling multimodal interactions (i.e., voice and touch-screen interactions) and discussing the benefits of visualizations as well as speech-interaction, students learn about the design recommendations, “G1: Visibility/Feedback of System Status” and “G9: Allowing Users to Recognize and Recover from Errors”. Applications that visualize agent conversations in addition to vocalizing them allow end users to better understand agents' goals and system status, and observe and recover from erroneous speech recognition.

Students also learn about the design recommendation, “G7: Flexibility and Efficiency of Use” in the ConvoBlocks activity. For example, students can allow agents to recognize entities at various times, which increases the the flexibility of the conversation. Students may program agents to recognize entities both during initial intent recognitions and in follow-up questions. Table 4.2 shows how the ConvoBlocks activity addresses these design guidelines.

4.2.3 Summary of the ConvoBlocks Analysis

Through the ConvoBlocks activity, students can learn the conversational agent concepts outlined in Table 4.4. They can also learn the conversational agent design best practices outlined in Table 4.2.

4.3 Activity 3: Convo

Convo is a voice-based system embedded in an online web interface allowing users to develop computer programs by conversing in natural language with a conversational agent. Teaching resources for the Convo activity can be found in Appendix G and the Convo code repository can be found on GitHub [204]. A programming interaction with Convo may go something like the following example. It involves a fictional student, Vanessa, who would like to teach her toddler nephew about different animal sounds.

```
[Vanessa opens up the Convo website, imagining a game where her nephew, Theo, says an
   animal name, and Convo repsonds with its sound.]
Vanessa: Convo, can you create a procedure?
Convo: What do you want to name the procedure?
Vanessa: Animal sounds
Convo: I created a procedure called Animal Sounds. What would you like it to do first?
[Vanessa pauses. She’d like to make the agent repeatedly ask for animal names, but she
    isn’t sure how. She notices an example phrase in the sidebar, though, that says
   "Create a loop", which sounds about right.]
Vanessa: Create a loop
Convo: Okay, when would you like the loop to end?
Vanessa: Until I say ’stop’
Convo: What would you like to do in the loop?
Vanessa: Get user input and save it as ’animal’
Convo: What would you like to do next?
Vanessa: If ’animal’ is ’dog’, play the dog sound
[Vanessa then proceeds to make conditionals for other animals, including cats, horses,
    and cows. She then closes the loop and finishes the procedure by saying, "Done".
```

Table 4.4: List of new conversational agent concepts the ConvoBlocks activity teaches, organized by concept category. Note that concepts in square brackets were taught in previous activities, but further built upon in this activity.

Category	**Concept**
Natural Language Understanding	Large language models
	Transfer learning
	Intents
	Entities
	Training
	Testing
Conversation Representation	Event-driven program representations
Dialog Management	Events
	Entity-filling
	Conditions
Data Access and Conversation Context	Contextual data
	Agent modularization
	Device access
	Cloud computing
Human-Agent Interaction	Text-based interaction
	Voice-based interaction
	Multimodal interaction
	[Speech synthesis]
	[Speech recognition]
	[Societal impact and ethics]

```
    Next, she opens up Convo's intent-training page and types in different phrases for
     a "get animal sound" intent, like "What do cows say?". She highlights animal
    names, like "cows", defining them as entity examples, similar to what is shown in
    Figure 4-17. Once she finishes, she returns to speaking with Convo.]
Vanessa: Connect the "animal sounds" procedure to the "get animal sound" intent.
[Satisfied with her program, Vanessa finds her nephew, Theo, and tells him to ask
    Convo about different animals.]
Theo: Dog noise
Convo: [Plays an audio file that barks excitedly]
Theo: Horse sound
Convo: [Plays an audio file with a long whinny]
[Theo and Vanessa laugh together as the whinny concludes.]
```

This system is similar to the Zhorai activity in how the interaction involves speaking with a conversational agent, like how "Morgan" and Zhorai interacted in Section 4.1. It is similar to the ConvoBlocks activity in its goal to empower users to develop new conversational agents, like how "Sheila" developed a talking storybook agent in Section 3.1. This tool aims to teach similar concepts to those taught with Zhorai and ConvoBlocks, and additionally dives deeper into how natural language processing occurs. In this section, I describe the platform and educational materials. Much of this information, as well as results from two studies with Convo and K-12 students, can be found in our papers [190, 215].

4.3.1 The Convo Platform

We designed the Convo system to support both voice- and text-based conversations. These conversations enabled three main tasks—program creation, program editing, and system feedback—through natural language. To illustrate, users might say, "Create a variable" and the system would reply, "What do you want to call the variable?", and so on, until the program is complete (e.g., see Figure 4-16). The user may then ask to execute the program or to go back and edit. If the user tried a command that was not understood, the system would respond appropriately (e.g., "I didn't quite catch that. What action did you want me to add?"). We describe more of Convo's abilities and example conversations in our papers [215, 190] and associated appendices [187].

The main additional concept the Convo activity introduces is how natural language processing can occur in different ways. Specifically, the activity introduces the idea of

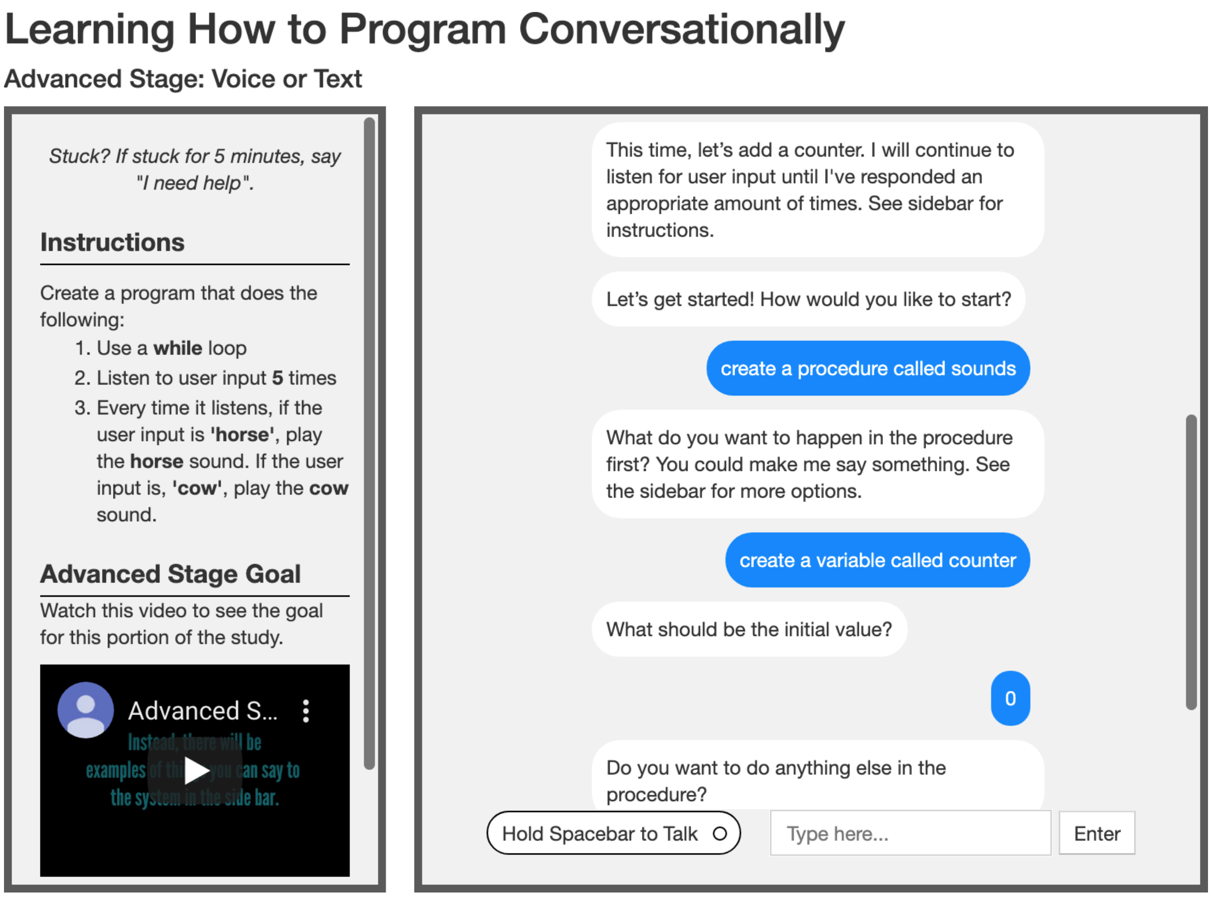


Figure 4-16: The advanced stage of the first user study with Convo. This page includes both a record button and text box for input [190].

constrained versus unconstrained natural language. With constrained natural language, users must say relatively specific preprogrammed phrases in order for the system to understand. In Convo's case, we implemented this with regular expression matching [35]. For example, with constrained natural language, if the user says, "Make a procedure", the system will match it to the expression, `"(?:make|create)(?: a)?(?:new)? (?:procedure|program)(?: (?:called|named)(.+))?\$"` [204], but if the user says, "Start a new procedure", the system will not recognize it.

With unconstrained natural language, users can say nearly anything to the system, and the system will attempt to classify what the user said into one of its programming actions. In Convo's case, we implemented this using a BERT-based classification model. For example, with unconstrained natural language, if the user says either, "Make a procedure" or "Start a new procedure", the system will classify them both into the `create_procedure` intent due to its pretrained language model that can generalize across phrases.

CONVO uses constrained (regular expression) natural language processing when users program conversational agents (as in Figure 4-16). The system uses constrained natural language here to reduce the potential for misclassification of phrases, since computer programming requires very precise language. CONVO uses unconstrained (machine learning, classification-based) natural language processing when end users are engaging with the agents users developed. This is due to the less precise, conversational nature of the developed agents. Users train these unconstrained, BERT-based agents using the GUI shown in Figure 4-17. As described in our paper [215], we developed curriculum for three-day workshops to teach students the trade-offs of constrained and unconstrained natural language, as well as other fundamental conversational agent concepts.

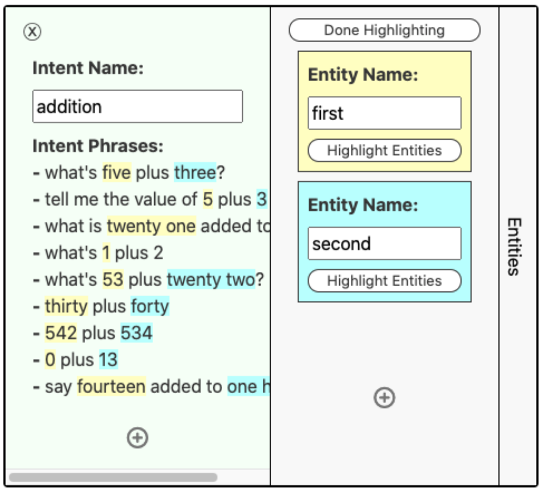


Figure 4-17: The CONVO GUI containing training data a user inputted to teach CONVO how to recognize when someone wants to add two numbers together. This training process is similar to the process in ConvoBlocks, which uses transfer learning. The phrases (left) are example inputs users can enter to trigger the "addition" intent. The two entities (right) are the two pieces of information (in this case, numbers) CONVO can learn to extract from intent phrases. CONVO is able to generalize across natural language using a BERT-based model, and recognize phrases that are not necessarily in the given intent phrases (e.g., "give me nine plus two") [215].

System Design

CONVO's natural language programming system consists of four system modules: the voice-user interface (VUI), natural language understanding module, dialog manager, and program manager, as shown in Figure 4-18. The VUI receives and transcribes voice input into text using Google's Cloud Speech-To-Text API [61] (if in a voice mode) and has a text box input (if in a text mode). The transcribed text is displayed on screen and sent to the natural language understanding module, which uses a regular expression-based semantic parser to determine intent and extract semantic information [190].

Once the natural language understanding module extracts the information, it sends the intent to the dialog manager, which keeps track of the conversation, system- and user-goals, and agent state; sends program-related information to the program manager; and generates appropriate responses. Convo shows the responses on screen and (if in voice mode) speaks them back to users using Google's Speech Synthesis API [60]. The program manager stores *actions* users specify for their program using a special program representation that can be converted to other formats (e.g., JavaScript, Python).

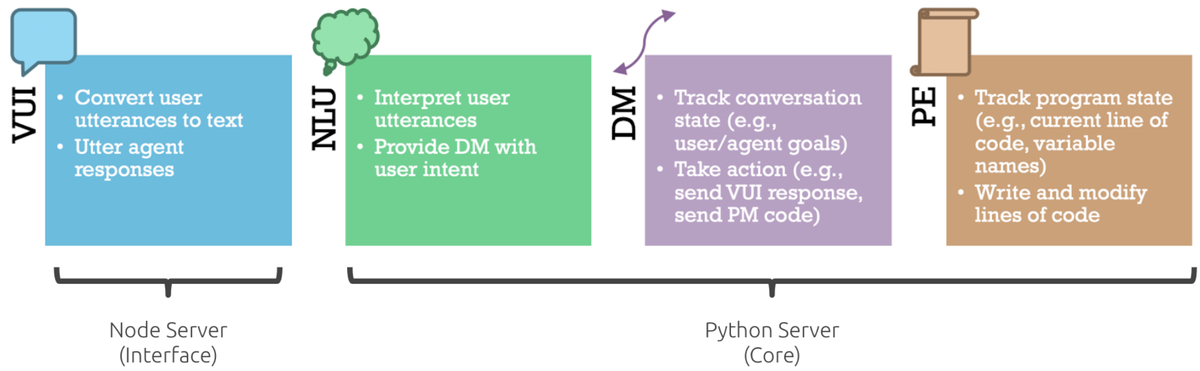


Figure 4-18: The four system modules Convo uses to create programs: the Voice User Interface (VUI), Natural Language Understanding (NLU), Dialogue Manager (DM), and Program Editor (PE) modules.

Users can train Convo to recognize phrases by providing a number of examples and performing transfer learning, similar to how they train agents in the ConvoBlocks activity. After this training process, users can then link programs they develop to the trained phrases. For example, users might train Convo to recognize phrases similar to, "What's five plus three?", and then link a program that computes addition to this phrase. When an end user says a similar phrase, Convo identifies the intent of the phrase (which in this case is linked to the "addition" program), and extracts entities (in this case, two numbers, like "five" and "three") using a BERT-based, fine-tuned large language model [215]. It then runs the associated program in Python using the entities extracted from the intent phrase. Figure 4-17 shows example training data for such an "addition" intent.

4.3.2 Convo Activity Analysis

Similar to the ConvoBlocks activity, the Convo activity consists of interactive lectures and group discussions about the "Big 5 AI Ideas" [178] and conversational AI; conversational agent development tutorials; and final projects. The Convo activity emphasizes different methods of natural language understanding, including constrained vs. unconstrained natural language. I describe the activity and the main conversational agent concepts involved in the curriculum in the sections below. More information about our design goals, development process and implementation of Convo, and the associated activity can be found in our papers and associated appendices [190, 187, 215, 216].

Lectures and Group Discussions

In this part of the activity, students listen to presentations by workshop facilitators and engage in group discussions. The presentations focus on how the "Big 5 AI Ideas" apply to conversational AI, how Convo works, and the differences between constrained and unconstrained natural language processing. Figure 4-19 shows a few of the presentation slides.

The Convo curriculum includes all of the natural language concepts developed in the ConvoBlocks activity (*large language models*, *transfer learning*, *intents*, *entities*, *training*, and *testing*). The main additional concept includes *constrained vs. unconstrained natural language*, as described below.

- **Constrained vs. unconstrained natural language** (Natural Language Understanding): In the *Convo* activity, students learn about how different types of natural language processing can be more appropriate for different applications. For example, when using *Convo* to program agents, intent recognition accuracy is very important. For instance, developers might become frustrated if *Convo* consistently misrecognizes "create a procedure" as "create a repeater", and every procedure becomes a loop. In this case, constrained natural language may be more appropriate than unconstrained. Constrained natural language generally has a higher recognition accuracy than unconstrained, as only pre-programmed, specific phrases are identified and recognized (e.g., through regular expression matching). On the other hand, with unconstrained natural language, the agent will attempt to recognize any phrase (e.g., through machine learning classification), but may not have as

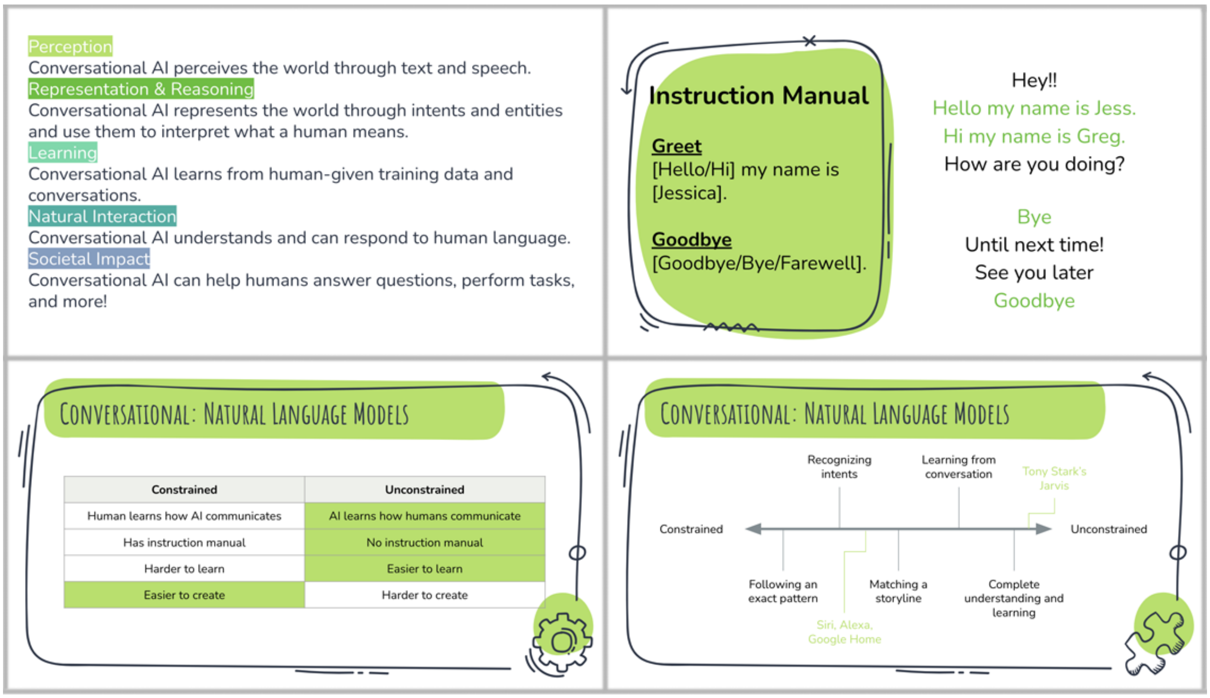


Figure 4-19: Slides from the CONVO curriculum. The top-left slide describes conversational AI in terms of the "Big 5 AI Ideas" [178]. The top-right slide introduces the concept of constrained natural language processing, and which sentences (in green) would be recognized by an constrained natural language "instruction manual". The bottom-left slide describes a number of advantages and disadvantages of constrained vs. unconstrained natural language. The bottom-right slide places various systems on a constrained vs. unconstrained spectrum. Additional slides can be found in Appendix G.

high recognition accuracy. Unconstrained natural language may be more appropriate for a social bot, as end users may value being able to say nearly anything to the bot over how accurately the bot understands. For instance, with a social bot, users may have more patience for an agent that misrecognizes their favorite food as "delicious falafels" instead of "chicken and waffles", since this does not have as significant consequences as when programming.

The concept of constrained vs. unconstrained natural language is related to a number of design recommendations, including, "G3: User Control and Freedom", "G5: Preventing User Errors", and "G7: Flexibility and Efficiency of Use". With constrained natural language, user freedom and flexibility of use decrease, as the system will only understand certain phrases. However, with an unconstrained natural language system, and with increased training, freedom and flexibility increase, as the system will attempt to understand nearly any phrase. As students add further training data and continue training their agents, their agents become better at understanding wide-ranging diction, preventing more user errors. Thus, through learning about constrained vs. unconstrained natural language, students can also learn about the G3, G5 and G7 design recommendations.

Tutorials and Final Projects

In the Convo activity, students complete two tutorials and develop final projects in groups. Students work in groups, since the Convo system has a limited number of GPUs for training. This introduces students to networking concepts, as well as hardware limitations in the cloud. Similar to the ConvoBlocks activity, the first tutorial involves training the system to recognize greeting and farewell intents. Students enter in training examples for both intents, as shown in Figure 4-20. They also program associated procedures by having a natural language conversation with Convo. Their programming enables Convo to respond with, "nice to meet you!", or "see you later", depending on the end user's given intent. An example of this natural language programming is shown in Figure 4-21.

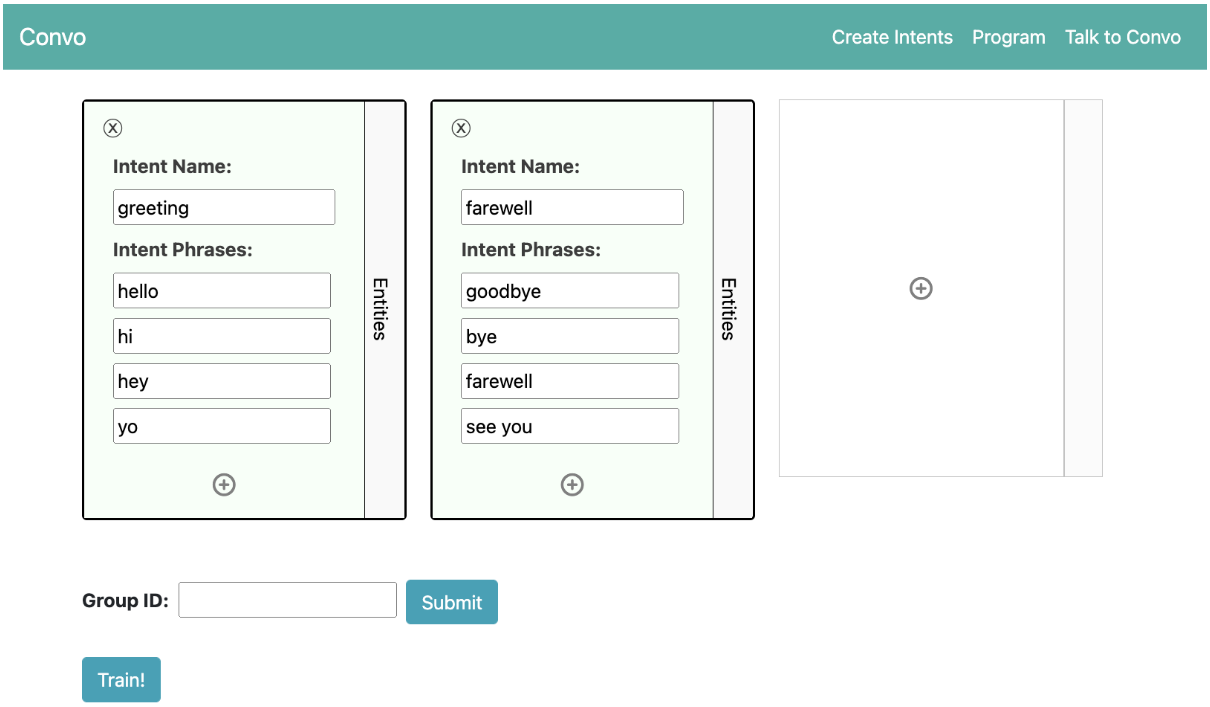


Figure 4-20: The Convo interface, showing two intents, including a greeting and farewell intent, and their associated training examples or "intent phrases".

For the second tutorial, students create a weather agent, in which end users can ask about the weather in different cities. Students accomplish this through using a location

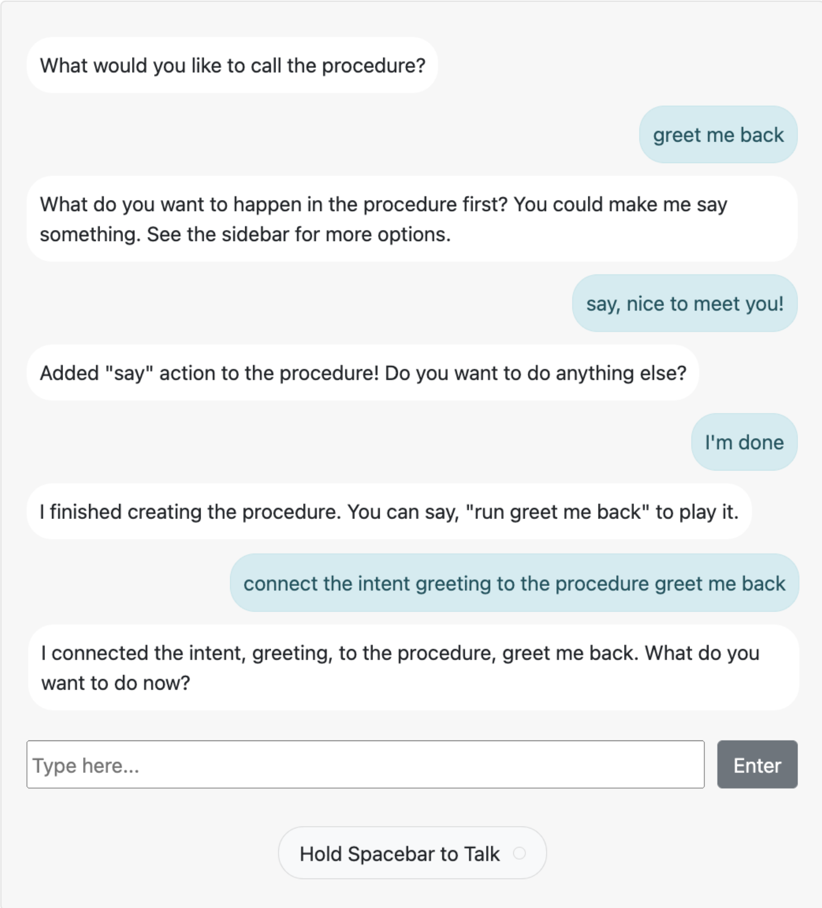


Figure 4-21: A natural language conversation with Convo, in which the user programs an agent to say, "nice to meet you!", and connects this procedure to the intent event named, "greeting". If an end user says something like, "hello" or "hi", the agent will respond with, "nice to meet you!".

entity, and an if-statement. The entity enables Convo to identify the city end users ask about, and the if-statement enables Convo to respond differently depending on the identified city. Figure 4-22 shows the intent with its location entity.

This part of the activity builds on a number of the concepts from the ConvoBlocks activity. It also introduces the concepts of *conversation state* and *task- vs. non-task oriented*, as described below.

- **Conversation state** (Dialog Management): When in programming mode, different events cause Convo to change state and answer developers' phrases differently. For example, if Convo is in its initial "home" state, and a developer says, "Create a procedure.", Convo would then begin creating a procedure and switch into its procedure-creation state. However, if Convo is already in its procedure-creation state, and the developer says, "Create a procedure.", it will respond with something

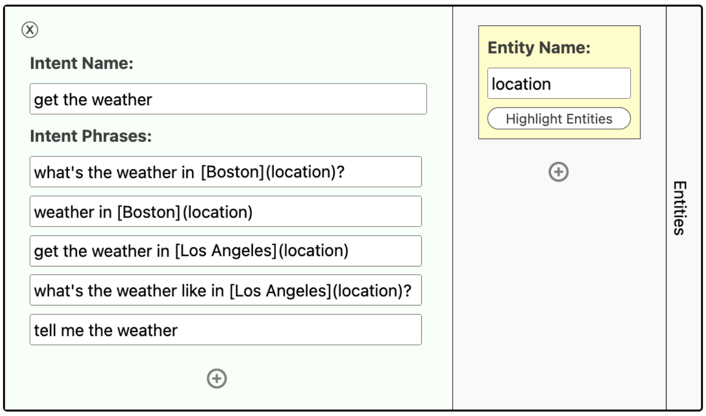


Figure 4-22: A weather-gathering intent with an associated location entity.

like, "If you'd like to create a new procedure, first exit this procedure and then create a new one", and will not begin creating a new procedure. Having different conversation states allow agents to take different actions in response to the same intents.

- **Task- vs. non-task-oriented** (Human-Agent Interaction): Often, agents (or their actions) are categorized as task- or non-task-oriented. For instance, when Convo is in programming mode it is engaging as a task-oriented agent, as all of its actions contribute to completing programming tasks, like creating procedures, if-statements, or loops. When Convo is executing developers' programs that are socially-oriented, like talking to end users about their favorite food, or asking them how their day is going, it is engaging as a non-task-oriented agent. Developers may want to make task-oriented agents' language more concise than non-task-oriented agents' language. They also may want to use constrained natural language processing rather than unconstrained.
- **Voice-based interaction** (Human-Agent Interaction): Students can interact with Convo by speaking or typing. Students may notice that speaking is generally more efficient than typing; however, spoken-input is generally less accurate than text-input [190]. Thus, when programming using Convo students may want to type; whereas when interacting socially with Convo they may want to speak.
- **Text-based interaction** (Human-Agent Interaction): Students can interact with Convo by typing or speaking. Students may notice that typing is generally more accurate than speaking; however, typed-input is generally less efficient than spoken input [190]. Thus, when programming using Convo they may want to type;

whereas when interacting socially with Convo they may want to speak.

- **Cloud computing** (Data Access and Conversation Context): In the Convo activity, students complete tutorials in groups, since Convo's computation resources are limited. Students learn that when developing agents, it is important to consider the amount of resources needed to train them. For example, if they do not have their own GPUs, or limited time using cloud-based GPUs, they may opt to develop simpler systems, like constrained natural language agents, instead of training-heavy unconstrained natural language agents.

Similar to Zhorai, the Convo agent itself implicitly teaches a number of the design guidelines through its design. For example, the Convo system records history conversation visually, allowing students to read it again later, if needed. This touches on the design guidelines, "G1: Visibility/Feedback of System Status" and "G6: Recognition Rather than Recall". Convo also addresses G6 through its sidebar, which shows example programming phrases students could say. This also addresses the guidelines, "G2: Mapping Between System and Real World", "G4: Consistency throughout the Interface", and "G10: Providing Help and Documentation", as it provides example conversation schema (G2), consistent conventions for speaking with Convo (G4), and help and documentation (G10).

When speaking with Convo, students may notice how it implicitly confirms actions it hears from students. For instance, a student might say, "Create a procedure", and Convo's response would be, "What do you want to happen in the procedure first?", confirming that it is creating a procedure. This addresses the design guidelines, "G5: Preventing User Errors" (since it ensures users know when Convo misrecognizes something), "G8: Minimalism in Design and Dialogue" (since the confirmation is implicit), and "G9: Allowing Users to Recognize and Recover from Errors". Finally, since students can converse with Convo using either voice or text, they can decide which input method is better depending on their context (e.g., in a loud coffee shop vs. in a quiet office). This addresses the recommendation, "A2: Considering How Context Affects Speech Interaction". Thus, Convo's design provides opportunities for students to learn a number of the design guidelines, as shown in Table 4.2.

Table 4.5: List of new conversational agent concepts the CONVO activity teaches, organized by concept category. Note that concepts in square brackets were taught in previous activities, but further built upon in this activity.

Category	**Concept**
Natural Language Understanding	Constrained vs. unconstrained natural language
Dialog Management	Conversation state
Data Access and Conversation Context	[Cloud computing]
Human-Agent Interaction	Task- vs. non-task-oriented
	[Text-based interaction]
	[Voice-based interaction]

4.3.3 Summary of the CONVO Analysis

Through the CONVO activity, students can learn the conversational agent concepts outlined in Table 4.5. They can also learn the conversational agent design best practices outlined in Table 4.2.

4.4 Activity 4: The Alexa Developer Console

The Alexa Developer Console is a website enabling developers to create Alexa-based agents. It targets developers who have programming experience, allowing them to code agents in JavaScript or Python, and train intent and entity recognition through a GUI, as shown in Figure 4-23 [14]. In this section, I describe the platform and a tutorial-based activity in which developers create a game-based agent. In future work, the concepts from this activity could be incorporated into the platforms and associated curricula for K-12 students, like those in the ConvoBlocks or CONVO activities.

4.4.1 The Alexa Developer Console Platform

The Alexa Developer Console allows developers to create agents similar to those developed with ConvoBlocks. They are event-driven, can be trained to recognize intents and entities, and run on Alexa-enabled devices. The main additional features include accessing webhooks and APIs, and deploying the agents to the Alexa Skills Store [14]. The web-based interface contains a GUI for training intent- and entity-recognition (shown

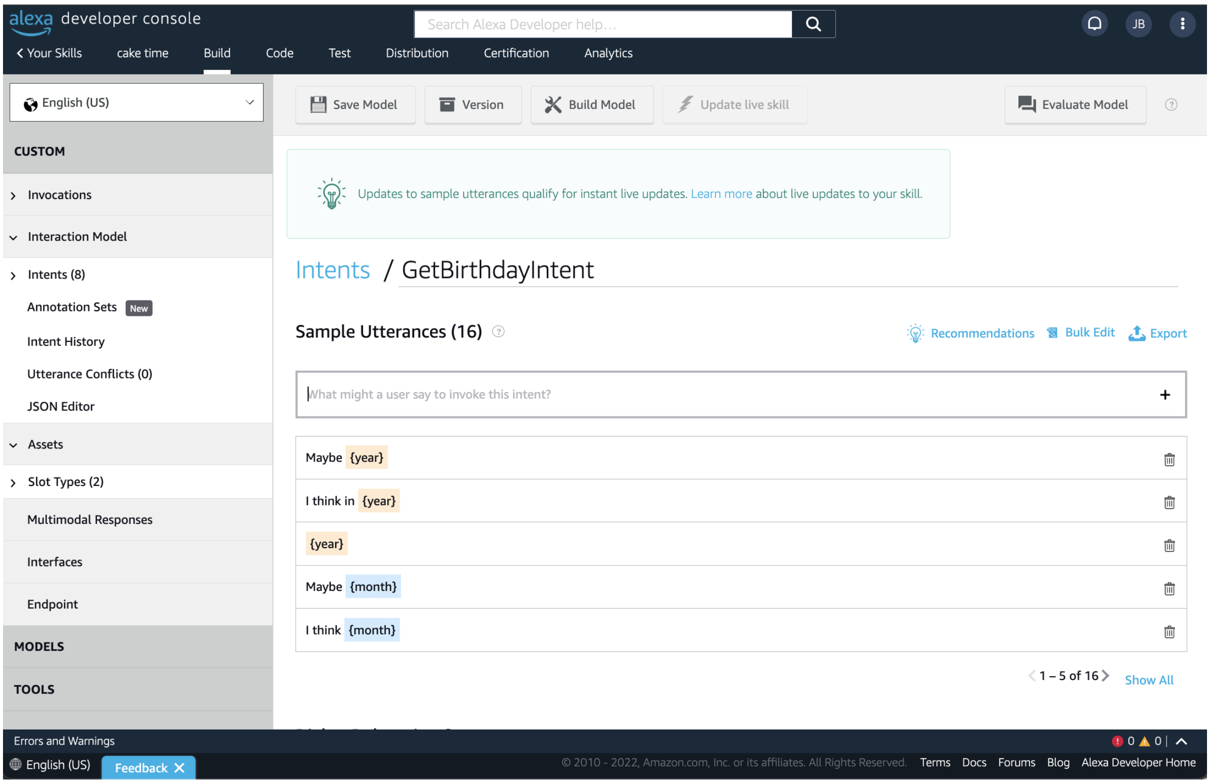


Figure 4-23: The Alexa Developer Console, showing training examples for an intent, including “year” and “month” entities [12].

in Figure 4-23), an IDE (shown in Figure 4-24), tools for defining other aspects of the agent, like multimodal responses and monetization, and pages for testing, distributing, certifying and analyzing the agent.

The general flow for creating an agent with the Alexa Developer Console involves defining intents and entities, programming responses through JavaScript or Python event handlers, testing the agent, and preparing it for deployment. Figure 4-25 shows how the agent processes and handles speech input. More information about the platform can be found in the Alexa Skills Kit [10].

4.4.2 Alexa Developer Console Activity Analysis

The Alexa Developer Console activity involves web-based tutorials in which developers learn to create a celebrity guessing-game agent called “Cake Time”, learn many conversational agent design best practices, and deploy their agent to the Alexa Skills Store. To illustrate the agent developed in this activity, an example interaction is below with a fictional end user named Joya, who has previously engaged with the agent.

`Joya: Launch Cake Time`

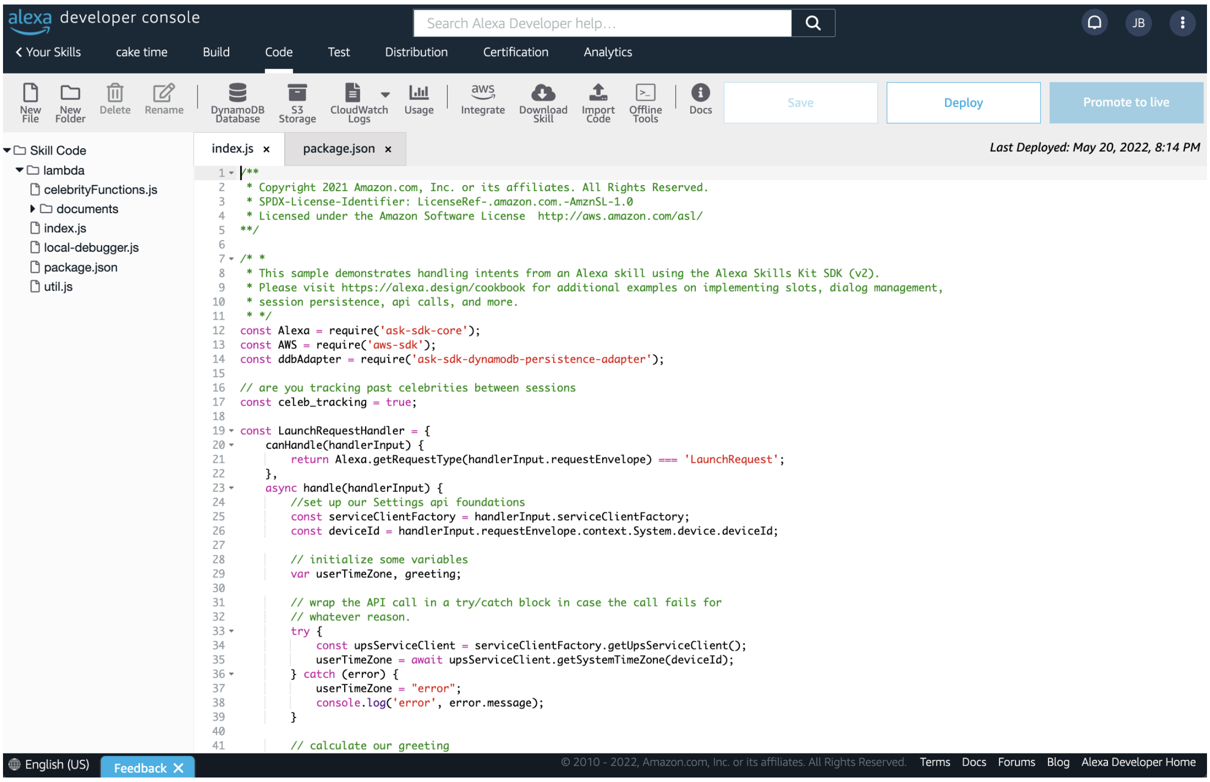


Figure 4-24: The Alexa Developer Console, showing JavaScript event handlers in the web-based IDE. Developers can alternatively use local IDEs and upload their code to the console [12].

Alexa: Hi night-owl! Welcome back to Cake Time! Ready to guess some more celebrity birthdays?

[A visual appears on the Alexa app with the words, "Say, 'yes.'", and "Play some Cake Time".]

Joya: Okay.

Alexa: In what month and year was Awkwafina born?

Joya: Hm, maybe around 1988?

Alexa: Thanks. What month were they born in?

Joya: She was born on my birthday, so in June!

Alexa: Yay! You got Awkwafina's birthday right! Your score is now 7. Want to try another?

[The words, "Congratulations!" and "Wanna go again?", appear on the Alexa app.]

Joya: No, I'm done.

Alexa: Goodbye!

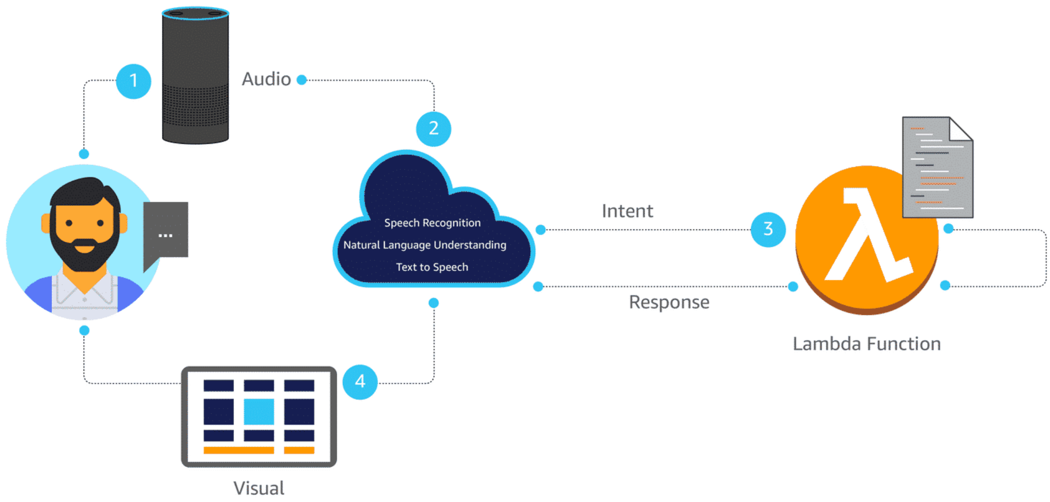


Figure 4-25: An illustration of how Alexa-based agents process information: (1) Through recording speech, (2) through speech recognition and natural language understanding models, (3) through event-handlers (often hosted on AWS Lambda), and (4) through speech synthesis and visual outputs. This diagram is from the Alexa Developer Console tutorial website [14].

Introduction

The introduction to the Alexa Developer Console activity consists of two main web pages. The first provides motivation for creating agents and an overview of how to develop agents. At a high level, it describes a number of the conversational agent concepts from the previous activities, including *agent modularization*, *events*, *device access*, *intents*, *speech recognition*, *speech synthesis*, *event-driven program representations*, and *testing*. It also develops the concepts of *webhooks and APIs*, *effective conversation design*, and *deployment*, which may have been hinted at in previous activities, but are discussed in detail in this activity. I describe these concepts below.

The second page emphasizes agent design best practices through developing the concept of an agent *storyboard*, showing video examples of poor agent design, describing well-designed agents, and listing design principles. This activity addresses nearly all of the design recommendations found in Table 4.2. For example, by describing the importance of enabling end users to cut through information hierarchies quickly, it addresses "G7: Flexibility and Efficiency of Use", "G8: Minimalism in Design and Dialog", and "G3: User Control and Freedom". It also addresses "G1: Visibility/Feedback of System Status" by describing how providing multimodal interactions (e.g., with touch, text, graphics, animations or videos) can provide useful feedback to end users. By emphasizing the importance of developing friendly, upbeat and helpful agent personas, it addresses, "DR-U2: Design conversational agent personas to foster appropriate partner models". It also describes how end users should always be able to use agents without these multi-modalities, addressing, "A2: Considering How Context Affects Speech Interaction", since end users' situations may prevent them from multimodal interaction. Table 4.2 shows the rest of the design recommendations the activity addresses.

The following list describes contextual examples of the main concepts touched on in the introduction.

- **Webhooks and APIs** (Data Access and Conversation Context): With the Alexa Developer Console, developers can add many features to their agents, including IoT connections, content feeds, music, and server queries. They can do so through APIs and webhooks. For instance, if a developer wanted to create an agent to control smart home devices, they could use the Smart Home Skill API, which enables inter device communication. Through the Alexa Developer Console activity introduction, developers learn about how APIs can enable agents to request information or other devices to take action. Later in the activity, developers learn how to access the Alexa Settings API to retrieve the current time and adjust agents' responses based on the results of the retrieval.
- **Effective conversation design** (Human-Agent Interaction): The introduction to the Alexa Developer Console activity recommends planning agent designs before writing code, outlines five main conversation design guidelines, and illustrates examples of effective and ineffective conversational agent designs. Developers learn how the five guidelines, "Stay close to Alexa's persona", "Write for the ear, then for the eye", "Be contextually relevant", "Be brief", and "Write for engagement to increase retention", can help them create more usable and engaging agents [14]. These guidelines align with many of the agent design recommendations from other literature, as shown in Table 4.2.
- **Storyboards** (Conversation Representation): To design effective conversational agents, the Alexa Developer Console activity recommends creating storyboards. Storyboards show a user's progression through dialog with an agent, including their responses, the contextual data the agent has stored, the agent's response, and any

questions the agents ask. For example, one storyboard frame might show how the agent knows the user's favorite song is "Never Gonna Give You Up" by Rick Astley, how the user stated, "Play my favorite song", and how the agent will respond with "Can you confirm that you'd like to be Rickrolled?". Storyboards are especially effective as design tools since they are user-centric and provide detailed information about the user's conversation, unlike flowcharts or graphs, for instance. Figure 4-26 shows an example conversation turn in an agent storyboard.

- **Deployment** (Human-Agent Interaction): In the introduction to the Alexa Developer Console activity, developers learn they can certify and publish their agents to the Alexa Skills Store. Unlike ConvoBlocks, which only allows developers themselves to access agents they designed, the Alexa Skills Store allows anyone to discover and interact with developed agents. Enabling anyone to use an agent comes with ethical and legal responsibilities. Thus, this process requires thorough distribution preparation and certification review, as described later in the activity.
- **Speech Recognition** (Human-Agent Interaction): The Alexa Developer Console activity provides additional detail about speech recognition, building on the concept first introduced in the Zhorai activity. Commercial systems, like Alexa- Siri- and Google-assistant-enabled devices, first listen for a "wake word", like "Alexa", "Hey Siri" or "Okay Google". Once the device recognizes the wake word, it starts recording users' commands. It then attempts to recognize the specific agent's unique identifier (i.e., the "invocation name", like "Favorite Song Player"), one of the intents associated with the specific agent (e.g., "Play my favorite song tomorrow at 9 AM"), and any additional entities (e.g., "tomorrow" and "9 AM"). Typically, the system performs the wake word recognition on the local device, whereas it performs the invocation name, intent, and entity recognition in the cloud.
- **Multimodal interaction** (Human-Agent Interaction): The Alexa Developer Console activity builds on the concept of multimodal interaction from the ConvoBlocks activity. For instance, developers can add visual text, graphics, interactive forms, animations, videos, and IoT interactions among other multimodal features, to augment the conversation. The activity materials note the importance of ensuring end users can use agents purely by voice; however, they also note how agents with multimodal (i.e., not purely voice) features are highly engaging [14].

- **Device Access** (Data Access and Conversation Context): The ConvoBlocks activity develops the concept of how end users can access agents on particular devices. The Alexa Developer Console builds on this idea and describes how agents on particular devices can access other devices. For instance, an agent running on the Alexa app could turn on a smart lamp, if the end user had given the agent access to the lamp through their Wi-Fi. Alexa-based agents can access devices through Wi-Fi, APIs, Bluetooth and other local and network connection methods [13]. Developers learn how they must program these types of connections into their agents, and end users must allow access to their devices for agents to access them.

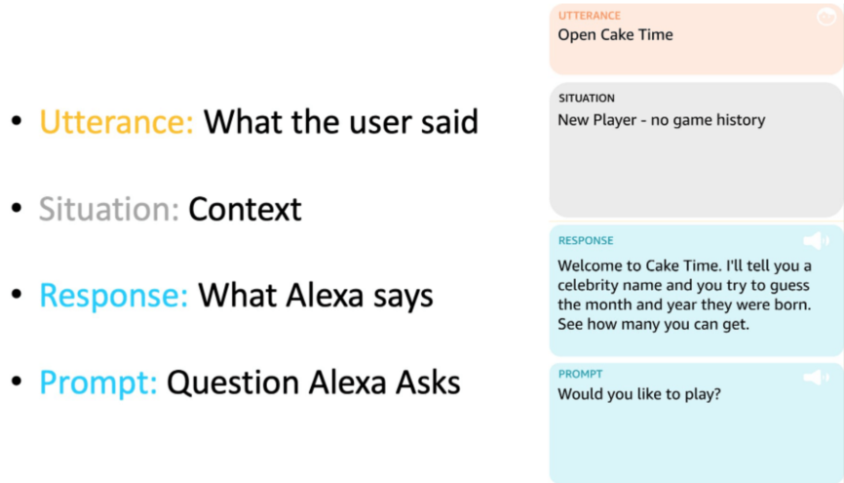


Figure 4-26: An example conversation turn from a storyboard in the Alexa Developer Console activity [14]. By combining multiple storyboard turn represenations together, developers can create a full storyboard to help envision how a conversation might progress.

Part 1: Agent-building Tutorial

In Part 1 of the Alexa Developer Console activity, developers begin creating a game-based agent. With this version of the agent, the end user begins a conversation, the agent asks them if they would like to play a game, and if they agree, the agent asks when a randomly selected celebrity was born. If the end user provides a month, but not a year, the agent asks for the year (and vice versa). Once the agent has both the month and year entities, it asks if the end user would like to guess again. In the next portion of the tutorial, developers enable the agent to determine whether the end user guessed correctly. Figure 4-27 shows an example conversation with the agent from Part 1.

In this portion of the activity, developers encounter many of the concepts developed

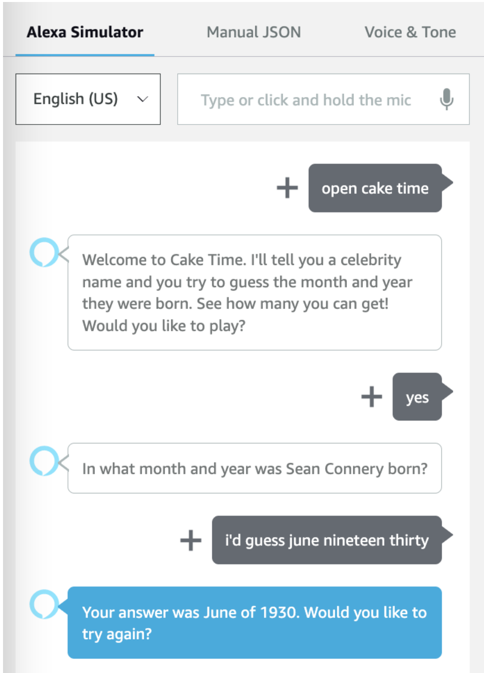


Figure 4-27: The Alexa Simulator on the Alexa Developer Console showing dialog from the agent developed in Part 1 of the tutorial.

in previous activities, including *cloud computing*, *agent modularization*, *intents*, *entities*, *event-driven program representations*, *training*, *testing*, *entity-filling*, *text-based interaction*, *voice-based interaction*, *multimodal interaction* and *events*. The following list describes the concepts which the activity extended further.

- **Event-driven program representations** (Conversation Representation): In the Alexa Developer Console activity, developers program event handlers in JavaScript or Python. This builds on the concept of event-driven program representations developed in the ConvoBlocks activity, in which the representation included visual blocks instead of text-based languages. In both cases, the structure of the representation includes an event definition (e.g., recognition of an intent) and a handler. The handler may include spoken, textual or multimodal responses; accessing webhooks or APIs; or computations, among other programmed reactions.
- **Training** (Natural Language Understanding): In the Alexa Developer Console activity, developers learn how to train agents for realistic conversations. The activity recommends including filler words in training examples, like “oh”, “like”, and “you know”, and other phrases not often included in formal writing, but often spoken. For instance, one useful (albeit informal) training example might be, “Okay, I think she was born in June of 1990”. Another might be, “I'd guess, like, June 1990”. By

training with many realistic examples like these, agents can become much more robust in their natural language understanding.

- **Entity-filling** (Natural Language Understanding): The Alexa Developer Console activity presents a method of entity-filling called, "auto-delegation". This method involves defining follow-up questions, which are automatically asked if an entity is unfilled. For example, if an agent needs to collect both a month and a year entity, and the end user only states, "1992", with auto-delegation, the agent would prompt the end user for the month (e.g., "And in which month?") before sending the entity data to the event handler.

Part 2: Memory and API Tutorial

In Part 2 of the Alexa Developer Console tutorial, developers learn how to store contextual data. For example, they enable their game-based agent to record end users' names, their scores, the number of times they have launched the agent, and which questions the agent has asked them. They also enable their agent to access the current time using an API. By the end of Part 1, the agent gives a different greeting depending on the number of launches and time of day, and checks to ensure it does not repeat any quiz questions.

This part of the activity extends the *contextual data* concept, as described below.

- **Contextual data** (Data Access and Conversation Context): In the Alexa Developer Console activity, developers learn about how there are three different types of contextual data. The first type, "request" data, is ephemeral and disappears after the agent completes the request or turn with the end user. For instance, one piece of request data might be the particular question the agent asked the end user, so it can correctly check the answer. The second type, "session" data, lasts for the entire session an end user is interacting with the agent, but disappears once the end user stops interacting. For instance, this might be a restaurant an end user wants to order from in one session (but does not necessarily want to repeat this order next time). The final type, "persistent" data, remains stored after the end user stops interacting with the agent. For instance, this might include the scores and names of multiple different end users, so that they can each access a leaderboard with scores from other sessions. With these three different types of data, agents can contextualize the conversation and recall long-term information while updating other information per-session.

This portion of the activity also addresses two design considerations: “G5: Preventing User Errors” and “G9: Allowing Users to Recognize and Recover from Errors”. Specifically, it addresses G5 through discussing how to prevent errors when end users provide unusual answers, like “Novemberish” for a month entity or a number longer than four digits for a year entity. For G9, the activity describes how developers can program their agents to ask questions to help the user recover from errors and rectify different situations. For example, if the end user has provided a month and year as an answer, but the agent has not provided a question yet, the developer can program the agent to ask something like, “I'm sorry, I'm not seeing an active question right now. Would you like a new question?”. Table 4.2 shows how the Alexa Developer Console activity addresses these design guidelines.

Part 3: Distribution Tutorial

In the final part of the Alexa Developer Console activity, developers learn how to deploy their agents to the Alexa Skills Store. They can do this publicly, only for specific business organizations, or as a beta test. They also learn the requirements for publication, how Amazon validates and tests their agents, and how they can roll-back their agents to previously certified versions, if the latest published version contains bugs. A description of the concept of deployment in this tutorial, extended from the introduction's description, is below.

- **Deployment** (Human-Agent Interaction): As mentioned in the introduction analysis, the Alexa Developer Console enables developers to publish their agents on the Alexa Skills Store. Later in the activity, developers learn the requirements for publication. This includes ensuring the agent meets privacy, security, policy, protected information, functional and other requirements. Through this activity, developers learn how deploying agents for public use holds them—as well as other stakeholders, like Amazon, for instance—to higher ethical and legal standards than when creating agents only for personal use.

When learning about the concept of deployment, developers set up privacy and compliance measures for their agents and provide end users with transparent descriptions about the agent, including whether it collects personal information, targets children,

Table 4.6: List of new conversational agent concepts the Alexa Developer Console activity teaches, organized by concept category. Note that concepts in square brackets were taught in previous activities, but further built upon in this activity.

Category	Concept
Natural Language Understanding	[Training]
Conversation Representation	Storyboards [Event-driven program representations]
Dialog Management	[Entity filling]
Data Access and Conversation Context	Webhooks and APIs [Device access] [Contextual data]
Human-Agent Interaction	Deployment Effective conversation design [Speech recognition] [Multimodal interaction]

contains advertisements, and contains items for purchase. This addresses the design recommendation, "A1: Ensure Transparency/Privacy", as shown in Table 4.2.

4.4.3 Summary of the Alexa Developer Console Analysis

Through the Alexa Developer Console activity, students can learn the conversational agent concepts outlined in Table 4.6. They can also learn the conversational agent design best practices from Chapter 2 and 3, as outlined in Table 4.2.

4.5 Activity 5: Dialogflow

Dialogflow enables developers to create agents for Google Assistant devices. Google has two main development platforms: Dialogflow ES [63], which is very similar to the intent based Alexa Developer Console platform, and Dialogflow CX [62]. Dialogflow CX provides a unique method of agent development using state machines, and allows developers to access advanced features, including what they call "flows" and "pages", as described in

Section 4.5.1 [62]. Note that due to the similarities between Dialogflow ES and the Alexa platform analyzed in Section 4.4, I only analyze Dialogflow CX in this dissertation to avoid repetitiveness. Also note that when I use the term, "Dialogflow", I am referring to Dialogflow CX.

To illustrate the final agent developers create in the Dialogflow activity, I provide an example conversation below, in which a fictional end user—and avid capture-the-flag player—orders a shirt.

```
[Elise, an enthusiastic capture-the-flag player, needs to order a yellow shirt for her
     championship game on Saturday. A player from the Red Hawks tore her last shirt
    while she was speeding into enemy lines, and she couldn't be a part of the
    Marigold Mavens without a trusty yellow t-shirt!]
Elise: Hi there
Dialogflow Agent: Hi! I'm an agent that orders shirts. How can I help?
Elise: Where is your physical store located?
Dialogflow Agent: You can find us at 484 Zorilla Street, Vancouver.
[Elise realizes this is quite a trek, and at 8 PM, she's not sure she'd make it in
    time for closing.]
Elise: What are your hours?
Dialogflow Agent: We're open until 9 PM on weekdays.
Elise: Can I order a shirt online?
Dialogflow Agent: Ok, let's start a new order. I'd like to collect a bit more
    information about the order. What size shirt?
Elise: Small
Dialogflow Agent: I'd like to collect a bit more information about the order. What
    color shirt?
Elise: Yellow
Dialogflow Agent: Ok, so you'd like a small yellow shirt. You can pick up your order
    for a small yellow shirt in two days. If you need to, you can cancel your order
    within 24 hours. Thanks for making a t-shirt order!
[Elise sighs with relief. Her next game isn't until Saturday, which is three days
    away. She would make it, and the Marigold Mavens would win the day!]
```

In the Dialogflow activity, developers create this agent through defining a state machine with intent based transitions, as described in the following sections. In future work, the concepts from this activity could be incorporated into the platforms and associated curricula for K-12 students, like those in the ConvoBlocks or Convo activities.

4.5.1 The Dialogflow Platform

The Dialogflow platform allows users to develop agents similar to those using ConvoBlocks, CONVO, and the Alexa Developer Console, as described in Sections 4.2, 4.3 and 4.4. These agents use natural language processing to classify intents and extract entities. Dialogflow additionally allows users to develop state machines to organize agents' conversational flow. These state-machine-based agents consist of basic building blocks called "flows" and "pages".

Flows define particular conversational paths end users might take. For instance a clothing store agent might have three flows: one for orders, another for returns, and another for exchanges. Pages define states in which agents are collecting particular information. For instance, the clothing store agent might have a page in the "order" flow for collecting information about the type of clothing the user wants (e.g., shirt, hat, shoes, etc.), another page for collecting information about the style of the piece of clothing (e.g., vintage, casual, bohemian, etc.), and a final page for collecting the size of the piece of clothing.

With agents developed using ConvoBlocks, CONVO, and the Alexa Developer Console, all intents and entities are top-level, meaning they can be accessed at any time. With Dialogflow agents, however, intents and entities are associated with specific pages and flows. This allows for scoping of the intents, as end users can only access them when the relevant flow is active and the relevant page is accessible, as shown in the example conversation in Figure 4-28. This can help reduce misclassification of intents, since there are fewer intents for the natural language processing module to choose from at any given time. Nonetheless, there is a tradeoff with this reduction of misclassification, as it may also reduce the flexibility of the system: End users may not be able to ask about a different topic in the middle of a conversation, if it is not included in the current flow. This increases the complexity of designing conversational agents, as developers must consider how to best scope the conversation.

4.5.2 Dialogflow Activity Analysis

Dialogflow's educational materials involve videos introducing the system, setup instructions, and an example tutorial showing how to develop a clothing-order agent [173]. I

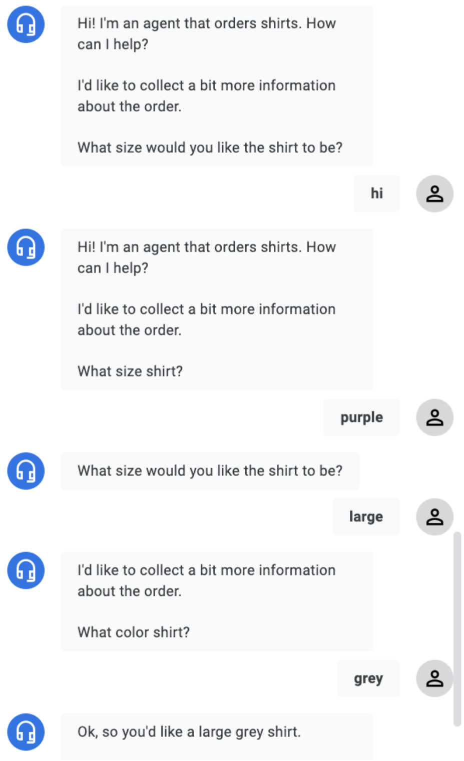


Figure 4-28: An example conversation in which an end user tests the clothing store agent that developers create in the Dialogflow tutorial. In the first turn, the agent tries to collect information about the size of a shirt, but the end user invokes the greeting intent. The agent then repeats its greeting and question about shirt-size, but the end user responds with the color of the shirt. The agent repeats its question again, and the end user provides the size. Finally, the agent asks about the color of the shirt and the end user responds with a color, and the agent confirms the end user's order. When the end user initially responds with a color to the agent's question about size, the agent does not recognize the color since the color intent was out of scope at that point in the conversation. This is due to flow and page modularization.

outline the major concepts taught through these materials throughout the following subsections and summarize them in Table 4.7.

Introductory Videos

Dialogflow's introductory videos are clearly intended for experienced software developers, as they highlight how the system allows for versioning, working across multiple teams, and advanced analytics. They also underscore how Dialogflow can facilitate the development of complex conversational agents. Without going into detail, the videos highlight concepts fundamental to developing Dialogflow agents, including "flows", "pages" and "state-based data models", in the context of an e-commerce agent. The setup and tutorial materials elaborate on these concepts.

The following list outlines specific conversational agent concepts mentioned in the introductory videos, provides contextual examples from the Dialogflow materials, and links the concepts to the conversational agent concept themes described in Table 4.8. I incorporate these concepts and examples into the conversational agent framework in Table 4.7.

- **State machines** (Dialog Management): Developers encounter state machines in Dialogflow when they create different "flows". These flows are depicted as directed graphs, or flowcharts with rectangles connected by arrows. The rectangles are the "pages" or conversational states in the state machines. As shown in Figure 4-29, one of the rectangles might represent an "Edit Order" conversational state, where the agent asks the user what type of pizza they would like. The arrows are the "transitions", which are typically triggered by user input; for example, when a user says, "I would like a pineapple pizza", the agent might transition to the "Confirmation" state, and respond with, "Are you sure you want pineapple on your pizza?". States and transitions are the building blocks of state machines, and are used in Dialogflow to manage conversational flow.
- **Flow and page modularization** (Data Access and Conversation Context): Developers encounter modularization of conversation with "flows" and "pages" in Dialogflow. Both flows and pages are reusable, compartmentalized pieces of conversation. Specifically, pages are conversational states or situations in which end users can respond with specific intents. In this way, pages encapsulate intents, preventing these intents from being accessed at any time in the conversation. If pages are specific situations in a conversation, flows are chunks of pathways in the conversation that connect these situations.

  Furthermore, flows encapsulate pages and the transitions between these pages, preventing pages from being accessed at any time in the conversation. For instance, if an end user was in the middle of a pet-store agent's flow about setting up a time to meet a cat, and they tried to access a dog-flow instead (e.g., by saying something like, "Actually, I want to know about dogs"), the agent would not understand. This is because—even though the agent would recognize this phrase in other contexts—the cat flow's scope does not include intent recognition for dog-related phrases. Thus, through preventing access to pages and intents at any time, and storing the current conversation state, Dialogflow scopes and manages conversational context.

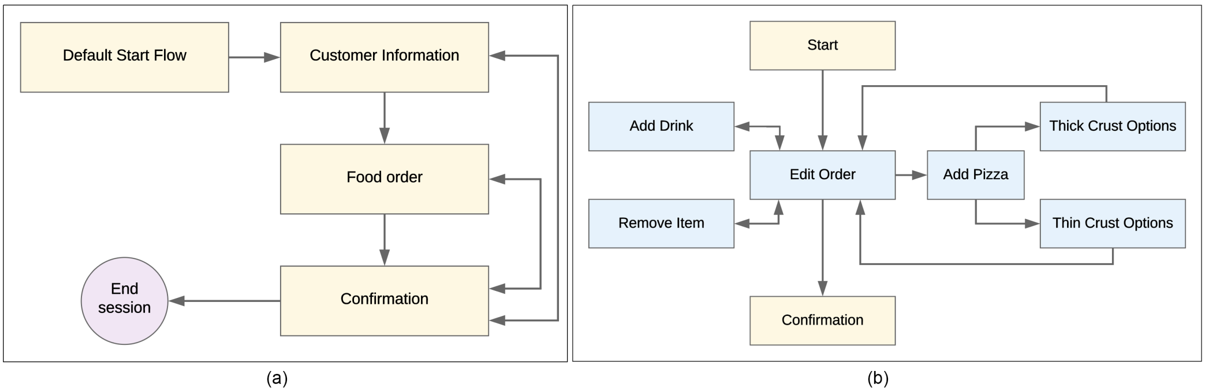


Figure 4-29: (a) An example representation of four flows, the "Default Start Flow", a "Customer Information" flow, a "Food Order" flow, and a "Confirmation" flow. (b) Within the "Food Order" flow, a user might develop pages (e.g., "Edit Order" page, "Add Pizza" page, "Add Drink" page, etc.). These diagrams are from Google's Dialogflow Guides [173].

Setup Instructions

The setup instructions again emphasize how the Dialogflow platform is intended for experienced developers. To begin, developers create a Google Cloud Project, set up billing, enable the Dialogflow API, and set up authentication for their project. This step highlights the cloud-based nature of current commercial conversational agents. In order for agents to be accessible, they must be stored somewhere accessible—ideally with constantly available computing resources—like the cloud. This builds on the concept of cloud computing first introduced in the ConvoBlocks activity (in Section 4.2). I describe a contextual example of cloud computing in the Dialogflow activity below.

- **Cloud computing** (Data Access and Conversation Context): In order to develop agents in Dialogflow, developers first set up a Google Cloud Project. This enables their agents to be run and accessed through the cloud. For example, a developer working for a clothing store might set up an agent on the Google Cloud and enable the clothing store's personal servers to communicate order delivery dates to the agent. Many current commercial agents, including Siri, Alexa, and the Google assistant, are deployed on the cloud. To communicate with local devices, the agent's code running on the cloud uses application programming interfaces (APIs), or special commands that both the cloud software and local device software understand. Developers encounter the concept of APIs in Dialogflow when setting up their Google

Cloud Project.

Agent Development Tutorial

The Dialogflow agent tutorial begins by directing developers to a page with definitions, including definitions for flows, pages, state handlers, entities, and intents, among other conversational AI related terms. After this, it guides developers through creating a clothing store agent. Figure 4-28 shows an example conversation with this agent.

The agent contains a single flow for gathering information about the store and ordering a shirt. The Dialogflow interface represents the agent being developed as a directed graph, as shown in Figure 4-30. Through creating a conversational agent in this environment, developers learn how to create new pages, associate intents, define entities, create transitions, and test agents. They also learn they can train their agent to recognize negative examples to reduce errors and enable conversation repair, among other conversational agent concepts. The following list describes contextual examples of main concepts touched on in the tutorial.

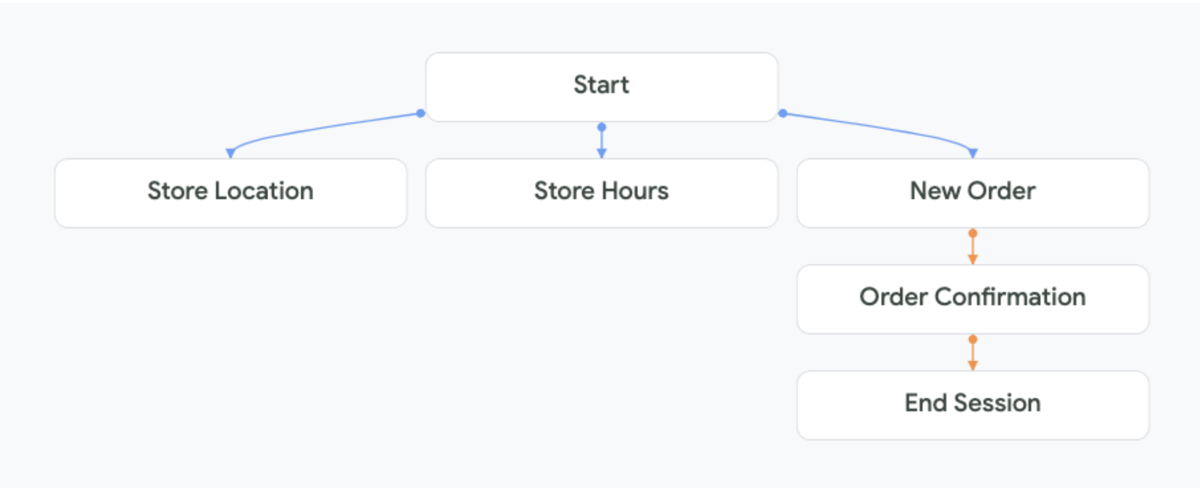


Figure 4-30: The final state machine or "flow" developed in the Dialogflow activity. The flow consists of transitions between five pages, including pages for gathering information about the store, and for ordering a shirt.

- **Training** (Natural Language Understanding): Developers encounter the idea of training agents on negative examples in the Dialogflow tutorial. For example, for a car-booking agent, developers might train the agent to recognize phrases like, "I'd like to buy a book about cars" (as opposed to a phrase like, "I'd like to book a car"), as a negative example or a phrase that should *not* trigger booking a car. By training incorrect examples of utterances, agents can learn to better understand users' intents.

- **Constrained vs. unconstrained natural language** (Natural Language Understanding): In Dialogflow, developers can use regular expressions, which is a constrained natural language technique, to capture entities. This is useful when entities have specific formats that are more difficult to recognize with unconstrained natural language techniques. For instance, a developer might want to capture a flight number, like "AA-484" or "AC-735", which can be easily described by the regular expression, `[A-Z]{2}-\d{3}`, but may be difficult for an agent to recognize through training.
- **Directed graphs** (Conversation Representation): Reading conversational agent code (like the event-driven code blocks that define agents in ConvoBlocks) can be confusing—especially when investigating complex agents containing lengthy conversational paths. Dialogflow represents conversational agents through directed graph representations. This representation emphasizes different conversational states, and how conversations can transition from one state to another. Figure 4-30 shows a directed graph representation of a clothing-store agent from the Dialogflow platform. In this example, there is a start-state, in which end users can ask about the store's location, about the store's hours, and to place an order. By asking one of these questions, the conversation transitions to the corresponding state (e.g., the "Store Location" state) where the agent can respond. Transitions are represented by one-way arrows and states are represented by rectangles.
- **Conditions** (Dialog Management): In Dialogflow, developers can add conditions to prevent or allow agents to transition to another conversational state. For example, if an end user is ordering a T-shirt from an agent, and the agent does not yet know the size of the T-shirt, a condition would prevent the agent from moving to the "order T-shirt" state until it received the required size information. Developers can implement other conditions too. For example, the agent may not be able to transition into the "order T-shirt" state until it knows there are T-shirts in stock via receiving a response from a webhook.
- **Deployment** (Human-Agent Interaction): Developers can deploy agents to many different platforms using the Dialogflow API. For example, developers could call the

Table 4.7: List of new conversational agent concepts the Dialogflow activity teaches, organized by concept category. Note that concepts in square brackets were taught in previous activities, but further built upon in this activity.

Category	Concept
Natural Language Understanding	[Training] [Constrained vs. unconstrained natural language]
Conversation Representation	Directed graphs
Dialog Management	State machines [Conditions]
Data Access and Conversation Context	Flow and page modularization [Cloud computing]
Human-Agent Interaction	[Deployment] [Effective conversation design]

Dialogflow API to create Twitter or Facebook Messenger bots, or Google Assistant voice-agents.

- **Effective conversation design** (Human-Agent Interaction): Although effective conversation design is not mentioned in the Dialogflow tutorial materials, Google provides secondary guides, including pages for "general agent design best practices" [174] and "voice agent design best practices" [175]. These guidelines address many of the design recommendations from Chapters 2 and 4. For example, they address "G2: Mapping Between System and Real World" by recommending the agent use speech patterns that enable effective speech recognition, since users tend to mimic agents [175]. Table 4.2 summarizes which recommendations the materials address.

4.5.3 Summary of the Dialogflow Analysis

Through the Dialogflow activity, students can learn the conversational agent concepts outlined in Table 4.7. They can also learn the conversational agent design best practices outlined in Table 4.2.

4.6 Summary of Concepts, Design Recommendations and Limitations

In this chapter, I outlined forty conversational agent concepts from agent development and interaction activities. These concepts aim to teach students how agents work and how to develop them effectively. They are categorized based on research in agent technology development [142, 33, 7], AI education [178, 100], and the five related activities described in this chapter [99, 185, 215, 14, 173], and were reviewed by the MIT Conversational AI for Kids research team.

I developed this portion of the framework as a starting point for teaching the fundamental concepts of conversational agent development. As research develops, other platforms emerge and conversational agent concepts evolve, researchers and educators can build upon and further develop this framework to better suit students' needs. The development platforms I chose to analyze in this chapter aim to cover a range of prior knowledge and understanding; nonetheless, future work includes analyzing further conversational agent activities outside of the five described in this chapter. Furthermore, although some of the activities include relevant studies and assessments analyzing students' abilities to learn these concepts, future work also includes developing additional assessments, and analyzing the comprehensiveness of the framework through further studies. This is especially the case for the concepts derived from the platforms intended for software developers, the Alexa Developer Console and Google's Dialogflow CX, as these have not been tested with K-12 students. Other future studies may include determining whether learning these concepts leads students to better calibrate their trust towards agents and conceptualize more accurate agent partner models.

In addition to addressing the concepts developed in this chapter, the activities addressed design recommendations from Chapter 2 and 3. I present comparisons of the different concepts and design recommendations addressed by the different activities in Table 4.1 and 4.2 respectively. As shown in the tables, in some cases, the activities addressed the concepts and recommendations directly, in other cases, they addressed them implicitly through agent design, and in other cases, they addressed them in materials separate from the curriculum. Using these tables, researchers and educators can determine which activities to refer to in order to teach students comprehensively about conversational agents. They also may refer to Table 4.8, which summarizes the concept categories and associated concepts.

Table 4.8: The conversational agent concept framework, which includes concepts from natural language understanding, conversation representation, dialog management, data access and conversation context, and human-agent interaction.

Category	**Concept**	
Natural Language Understanding	Semantic analysis	Intents
	Machine learning	Entities
	Similarity scores	Training
	Large language models	Testing
	Transfer learning	Constrained vs. unconstrained natural language
Conversation Representation	Textual representation	Event-driven program representations
	Undirected graphs	Storyboards
	Histograms	Directed graphs
Dialog Management	Turn-taking	Conditions
	Events	Conversation State
	Entity-filling	State machines
Data Access and Conversation Context	Pre-programmed data	Device access
	User-defined data	Cloud computing
	Contextual data	Webhooks and APIs
	Agent modularization	Flow and page modularization
Human-Agent Interaction	Speech synthesis	Voice-based interaction
	Speech recognition	Multimodal interaction
	Recovery	Task- vs. non-task-oriented
	Societal impact and ethics	Deployment
	Text-based interaction	Effective conversation design

Chapter 5

Recommendations for Teaching Conversational Agent Concepts and Design to K-12 Students

Technology is just a tool.
In terms of getting the kids to
work together and motivating them,
the teacher is the most important.
—Bill Gates

In Chapter 3, I discussed how it is important to consider how K-12 students perceive agents, and how we can empower them to calibrate their perceptions and trust in healthy ways. In Chapters 2 and 4, I outlined specific design guidelines and concepts that can be taught to students to help them understand and calibrate their perceptions in this way. In this chapter, I outline how educators and researchers can best implement these design guidelines and concepts in the K-12 classroom. I do this through the teaching recommendations shown in Table 5.1. These recommendations are tailored to the unique needs of K-12 students and the learning goals of the conversational agent design guidelines and concepts.

As shown in Table 5.1, each recommendation is especially relevant to teaching a number of the conversational agent concept categories. For example, the recommendation to utilize data relevant to students' lives (T3) is especially applicable to teaching the concepts within the Data Access and Conversation Context category. In this Chapter, I

Table 5.1: This table outlines which teaching design recommendations (left) are particularly relevant to which conversational agent concept categories (top), represented by a filled square (■).

	Natural Language Understanding	Conversation Representation	Dialog Management	Data Access and Conversation Context	Human-Agent Interaction
T1: Incorporate High Iterativeness and Instant Feedback	■		■	■	
T2: Empower Students to Learn by Teaching	■			■	
T3: Utilize Relevant Data	■			■	■
T4: Utilize Gamification and Embodiment		■	■		
T5: Integrate Conversational Agent Concepts into Core Curricula		■			■
T6: Reduce the Prior Experience Gap			■		

describe the recommendations with respect to the categories particularly relevant to them (as outlined in the Table), although they may be useful for teaching the other categories as well. Furthermore, the recommendations are grounded in research from a systematic literature review of K-12 AI education, which I completed alongside Zhou and Lin [214]; an AI curricula co-design study with K-12 teachers, which I completed alongside Lin [188]; and studies in which my co-authors and I taught conversational agent development and concepts to K-12 students [99, 185, 193, 191, 215, 190]. Thus, the recommendations are particularly relevant to the K-12 context; however, they may be useful for teaching at the university level or other contexts as well.

5.1 Teaching Natural Language Understanding

In this section, I outline how three teaching recommendations (T1, T2 and T3) can facilitate learning Natural Language Understanding concepts. As described in Chapter 4, the Natural Language Understanding category teaches students how agents can analyze human language and derive meaning from it.

5.1.1 Natural Language Understanding Requires Iteration

One general theme connecting a number of the concepts in the Natural Language Understanding category includes how large amounts of data are required in order to create effective Natural Language Understanding models. For instance, for a model to recognize *Intents* effectively, many variations on *Training* phrases are required. If students want to train an agent to recognize when someone wants to hear an inspirational quote, for instance, some training phrases may include "Give me an inspirational quote", "Share some wisdom with me", and "Tell me a quote". Typically, if a *Large language model* has already been developed, and users are utilizing *Transfer learning* to enable agents to recognize intents (as in the ConvoBlocks [193], Convo [215], Alexa Developer Console [12] and Dialogflow [62] activities), around 25 training phrases are needed, as a good rule of thumb [77]. Furthermore, it is important to perform thorough *Testing* of the system to ensure its effectiveness. To facilitate thorough testing and development of large amounts of training data, one teaching recommendation includes, **T1 Incorporate high iterativeness and instant feedback**.

In the context of the Zhorai activity, this recommendation may involve encouraging students to provide many training sentences about animals, testing how well Zhorai classifies these animals into ecosystems, and then incorporating additional training sentences (i.e., "iterating") until Zhorai's classification is satisfactory [99]. (See Figure 5-1.) Similarly, with ConvoBlocks and Convo, students must iterate on their intent classification models by inputting additional training phrases until the models can adequately recognize intents [215, 191]. By allowing these systems to make mistakes, and having students engage in experimental learning through iteration, students can reassess—and reinforce—their understanding of Natural Language Understanding concepts [218, 99]. Furthermore, by incorporating instant feedback, like how students can instantly tell whether Zhorai has made classification mistakes, students can immediately begin to reason about the system and continue in this concept-reinforcement process [209, 99].

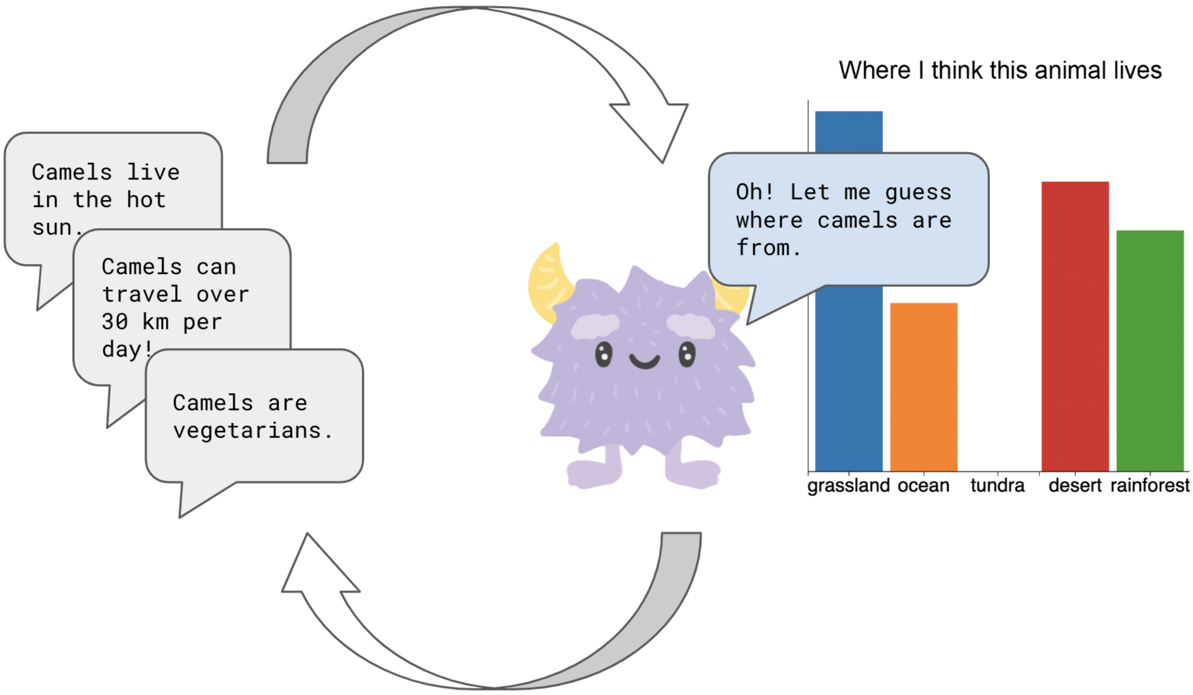


Figure 5-1: Through iterating on the training sentences provided to Zhorai, students can develop a better classification model, and reinforce the concepts of *Training*, *Testing*, *Machine learning* and other Natural Language Understanding concepts.

5.1.2 Natural Language Understanding Relates to Student and Language Learning

In general, the concepts in the Natural Language Understanding category relate to how agents can derive meaning or useful information from phrases. These concepts range from how agents can classify parts of speech, to how agents can find similarity between words, to how agents can learn about language through data. Because these concepts all relate to agents understanding or learning language, they lend themselves to the learning-by-teaching paradigm. As described in Section 1.2.3, this method involves students teaching others, and through this process, learning about the concepts they are teaching. In this way, students can teach agents different information and learn concepts including *Machine Learning* and *Training*, as in the Zhorai activity [99], or teach agents to recognize different phrases and learn concepts including *Intents* and *Entities*, as in the CONVO activity [215].

This method of learning-by-teaching not only can improve student outcomes, but can also increase engagement, which is especially important in K-12 settings [36]. Increasing engagement can improve throughput rates, retention of students and equality [180]. Through our (Zhou et al.) literature review, we found learning-by-teaching to be an effective method when teaching AI-related content, like conversational agent concepts; hence the teaching recommendation, **T2 Empower students to learn by teaching** [214].

5.1.3 Natural Language Understanding Requires Large Amounts of Data

Since language learning often requires large amounts of data (as discussed in Section 5.1.1), Natural Language Understanding concepts also lend themselves to the teaching recommendation, **T3 Utilize relevant data**. For example, students could learn about *Large language models*, *Training* and *Similarity scores* through investigating whether a language model was trained on gender-biased data. One classic example of this is a language model activity from MIT's Deep Learning Practicum course [93] in which students learn how certain models associate terms like "computer programmer" to be more similar to terms like "male" than to "female" due to a biased training data [93, 27]. In our literature review of K-12 AI education research, we found that utilizing data directly relevant to the students themselves (like gender-biased data); concrete, contextually-relevant data; or data to solve real-world problems when teaching can be especially engaging [75, 171, 170, 169, 154, 48, 186, 153, 214]. It can also increase students' confidence in being able to make an impact in their communities, as indicated by the results from Chapter 3.

5.2 Teaching Conversation Representation

In this section, I outline how two teaching recommendations (T4 and T5) can facilitate learning Conversation Representation concepts. As described in Chapter 4, the Conversation Representation category teaches students how to model agent dialog to facilitate the agent development process.

5.2.1 Conversation Representation Can Be Embodied and Gamified

As developers create agents, they need methods to readily perceive the flow of potential human-agent dialog and language. This may involve creating *Storyboards* to provide thorough conversational context for human-agent dialog, as in the Alexa Developer Console activity [14], or *Directed graphs* to show dialog alternatives, as in the Dialogflow activity [173], as two examples. To create these representations, developers must consider how to transform actual conversations into graphics or textual representations, which can then easily be transformed into computer representations (e.g., *Event-driven program representations*). Oftentimes, this transformation process can greatly benefit from engaging with other humans in the conversations being transformed, as when the conversationalists voice the dialog, they can easily identify awkward phrases or unnatural interactions. At this point in the agent development process (which is typically early on) this is especially helpful, since dialog can still be easily modified before it is implemented as a computer representation. This embodiment of conversation leads to the recommendation, **T4 Utilize gamification and embodiment**.

Embodiment can be described as perspective-taking, and is oftentimes used to engage students in learning AI concepts [186, 75, 72]. Gamification often goes hand-in-hand with embodiment and AI learning, and can be described as transforming learning activities into play [153, 131]. For instance, to gamify a Conversation Representation development process, students might draw *Directed graph* (or other) representations of proposed agent dialog, such as the graph shown in Figure 5-2; role-play as the agent and end users, following along with the proposed dialog flow; and try to find situations in which the dialog gets "stuck" or does not make sense. When end user players find such situations, they score points. After so many turns, whoever has the most points wins. Both embodiment and gamification, as in this example, can increase student engagement and learning outcomes [153, 214, 131].

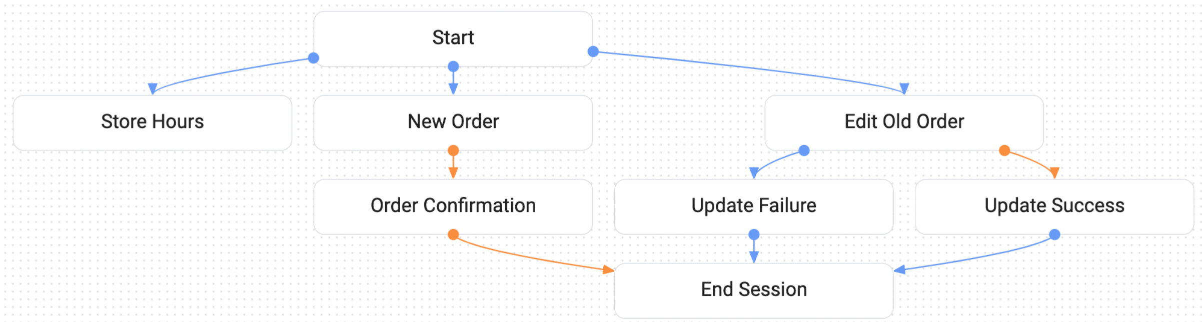


Figure 5-2: An example directed graph, representing a clothing-ordering agent. To engage students in learning about *Directed graph* representations, students may engage with such graphs in a game to identify situations in which the agent would get "stuck". For example, students might role-play as agents and end users. If the agent player is currently in the Update Failure state (middle-right), and the end user player asks for the store hours, the agent would not be able to respond, since the Store Hours state (left) is disconnected from the Update Failure state. The end user player would score a point for finding this situation, and the students could update their graph to allow the agent to access the Store Hours state in this situation.

5.2.2 Conversation Representation Includes Concepts from Other Subjects

As students develop Conversation Representations, they can also learn concepts from other core K-12 subjects. For instance, as they role-play agent conversations, they can learn drama skills, or as they create *Textual representations*, they can learn written language skills. This can be especially helpful in the K-12 education context, as oftentimes teachers cannot teach subjects outside of their core curricula [188, 214]. This leads to the recommendation, **T5 Integrate conversational agent concepts into core curricula**. Another example of core curricula integration includes the Zhorai activity's *Histogram* representations of *Similarity scores* [99]. Since *Histograms* are often used in math and science courses, teachers can utilize the Zhorai activity to teach these subjects while also teaching Conversation Representation concepts.

5.3 Teaching Dialog Management

In this section, I outline how three teaching recommendations (T1, T4 and T6) can facilitate learning Dialog Management concepts. As described in Chapter 4, the Dialog Management category teaches students how agents sequentially decide what to say in response to input, including human speech.

5.3.1 Gamification and Embodiment (T4) also Apply to Dialog Management

In many ways, Dialog Management concepts are similar to those from the Conversation Representation category. Thus, similar teaching recommendations can be implemented to foster learning in both categories. Dialog Management concepts, however, focus on the details of how agents decide what to say next, rather than how developers organize dialog development. With respect to the gamification (T4) example presented in Figure 5-2, in which students learn about *Directed Graphs*, students can additionally focus on learning how and why the agent makes particular dialog decisions. These decisions might have to do with which *Conversation state* the agent is in, whether a particular *Condition* has to be met to move into another state, or how developers designed the flows within the agent's *State machine*. To illustrate, students may have designed a pet simulation agent with the *State machine* represented in Figure 5-3. If students engaged in a role-playing game with this agent, and were in the "Feed pet" *Conversation state*, they would need to make a dialog decision depending on the "Hungry?" *Condition* of the virtual pet. By embodying this agent, and making such decisions themselves, students can learn Dialog Management concepts, like *State machines*, *Conditions* and *Conversation states*. As described previously, this type of embodiment and gamification can be especially engaging for K-12 students [105, 206, 199, 152, 131, 53].

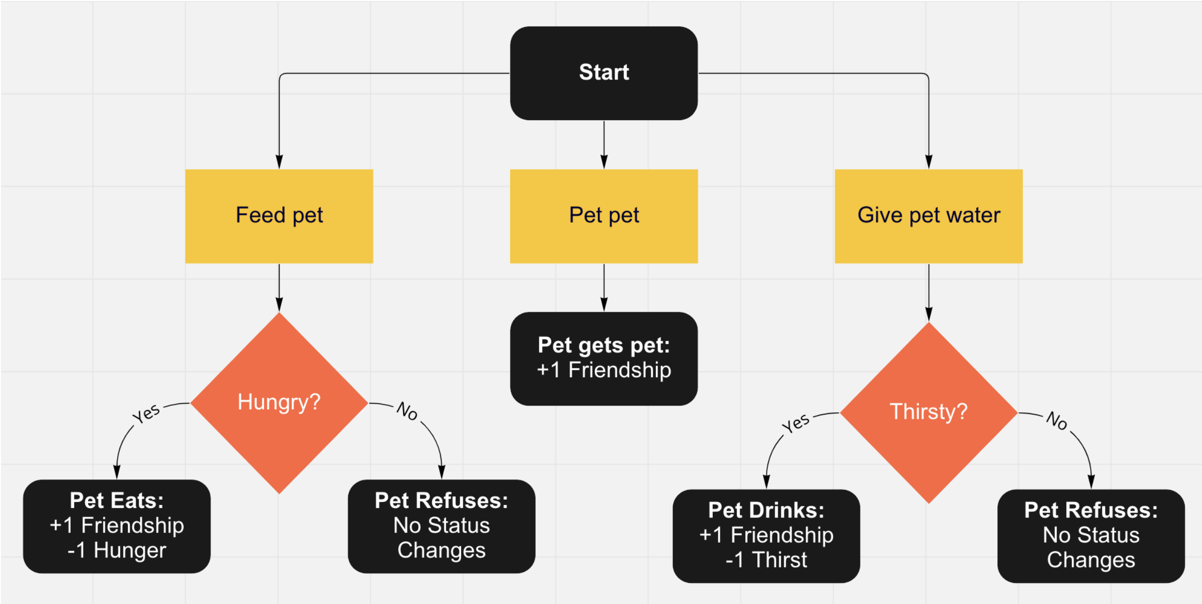


Figure 5-3: Students may develop a virtual pet agent, as in this *State machine* represented by a *Directed graph*. This representation identifies *Conditions* using orange diamonds and *Conversation states* using yellow and black rectangles.

5.3.2 Dialog Management Involves Understanding Computer Programming

Dialog Management systems are generally highly programmatic. For example, *Turn-taking*, or the back and forth nature of user input and agent output in agent development, is a step-by-step process. This mirrors the computer programming concept of "Sequences", or how programs can be described by series of steps [31]. For students who have prior programming experience, the concept of *Turn-taking* will likely be intuitively understood; however, for those who have never encountered programming before, it may be less naturally understood. For other, more advanced concepts highly related to programming, like *Events*, *Conditions* and *State machines*, the gap between how easily students understand them (depending on prior programming experience) will likely be more pronounced. Since K-12 students often have varying amounts of prior experience, it is important to consider how to mitigate this gap. Thus, I present the recommendation, **T6 Reduce the prior experience gap**.

To overcome this challenge, educators may use low-barrier-to-entry, high-ceiling conversational agent activities, like block-based coding with ConvoBlocks [191, 83, 50]. Block-based coding bridges gaps in experience by eliminating syntax errors while still enabling students to develop complex, industry-level programs (including agents deployed on the Alexa ecosystem). Other opportunities to reduce experience gaps between students include providing starter codes, such that students can learn from observing previously developed agents, as well as build on them; and providing detailed agent development tutorials, which students can follow along at their own pace [65, 191]. Students may also have varying levels of prior experience interacting with conversational agent systems, as shown in the study in Chapter 3. Thus, it can be helpful to provide opportunities for all to interact with such systems as well to reduce prior experience gaps.

5.3.3 Iteration and Instant Feedback (T1) also Apply to Dialog Management

Programming involves copious amounts of iteration. Even for highly experienced software engineers, it is extremely rare to develop a program—or even a code snippet—without encountering a bug and having to iterate. Developing conversational agents is not an exception to this, but rather requires perhaps even more iteration than other programming projects. *Entity filling* models can be poorly trained; *Events* can cause unwieldy levels of callbacks; *State machines* can result in confusing modularization, preventing top-level access to intents. Instant feedback, as described in Section 5.1.1, can help to ensure students do not give up after encountering bugs. Furthermore, since developing Dialog Management systems necessarily involves iteration, the recommendation of utilizing iteration to reinforce concepts (T1) is especially applicable.

5.4 Teaching Data Access and Conversation Context

In this section, I outline how three teaching recommendations (T1, T2 and T3) can facilitate learning Data Access and Conversation Context concepts. As described in Chapter 4, the Data Access and Conversation Context category teaches students how agents send, receive and store information to use in conversation.

5.4.1 Utilizing Relevant Data (T3) also Applies to Data Access and Conversation Context

The foundation of this category of conversational agent concepts is data: What type of data agents use, and how they use this data. As discussed in Section 5.1.3, to engage K-12 students in learning experiences, it is often helpful to use data relevant to students' lives or to real-world problems (T3). For example, students might train speech recognition models with their own, *User-defined* speech *data*. They may also utilize *Webhooks and APIs* to create agents to provide up-to-date information about climate change. This not only enables them to learn Data Access and Conversation Context concepts, but it empowers them to develop socially useful and relevant agents.

5.4.2 Iteration and Instant Feedback (T1) also Apply to Data Access and Conversation Context

A number of the concepts in the Data Access and Conversation Context category can quickly become complex as agent programs grow. For instance, as developers add more flows and pages to their agents, it can become difficult to know which flows or pages are accessible, or where *modularization* of *agents*, *flows and pages* begin and end. In order to determine what is accessible in particular conversation contexts, it is important to provide students with instant feedback about their project—especially at the K-12 level (T1). This can help sustain engagement and improve learning.

5.4.3 Empowering Students to Learn by Teaching (T2) also Applies to Data Access and Conversation Context

One of the key themes in the Data Access and Conversation Context category is how data used by conversational agents can come from different locations. For instance data may come from other devices (*Device access*), the cloud (*Cloud computing*), web sources (*Webhooks and APIs*), the programming itself (*Pre-programmed data*) or even end users (*User-defined data*). To facilitate understanding of this theme, students can experimentally interact with the agent to determine what the agent "knows" (i.e., which data it can access). Utilizing the learning-by-teaching paradigm (T2), students can also provide the agent with *User-defined data*, as in the Zhorai activity, and observe how this affects what it "knows". Through this paradigm, students can learn how agents access data, while taking part in an engaging, effective learning method [99, 214, 180].

5.5 Teaching Human-Agent Interaction

In this section, I outline how two teaching recommendations (T3 and T5) can facilitate learning Human-Agent Interaction concepts. As described in Chapter 4, the Human-Agent Interaction category teaches students about the interface between agents and humans, and how developers can create this interface to be ethical, usable and effective.

5.5.1 Utilizing Relevant Data (T3) also Applies to Human-Agent Interaction

The concepts in this category emphasize the human aspect of conversational agent design. For instance, *Recovery* involves interacting with end users to resolve misunderstandings or errors, *Speech recognition* involves recognizing human speech, and *Societal impact and ethics* involves determining whether agents affect humans in generally healthy and positive ways. Educators can teach each of these concepts with respect to data relevant to the humans they are teaching. For instance, educators could teach *Recovery* and *Speech recognition* to students through identifying when agents make recognition errors at higher rates. They could ask students whether the agent makes more mistakes when interacting with different subsets of people than others (e.g., people with higher vs. lower voices, or with various accents), and have students use their own speech data to determine the answer. This same example could play a role in teaching *Societal impact and ethics* by asking students whether it is ethical if agents understand certain subsets of people better than others, and how this might be changed using different training data. By doing so, educators can engage students through utilizing relevant data (T3) when teaching Human-Agent Interaction concepts.

5.5.2 Integrating Conversational Agent Concepts into Core Curricula (T5) also Applies to Human-Agent Interaction

As with the concepts in the Conversation Representation category, many of the concepts in the Human-Agent Interaction category are highly relevant to core subjects in K-12 education (T5). For instance, *Societal impact and ethics* are directly related to social studies and history curricula. Students can investigate how other technologies have affected society in the past, while relating to how conversational agents could affect today's society. Furthermore, *Voice-based interaction* is highly related to drama, and *Text-based interaction* is highly related to written language courses. Through integrating conversational agent concepts into core curricula, students may find learning these concepts easier when they learn them in subjects they are already familiar with (especially initially, when they are not familiar with learning about agents), and educators may find it easier to justify teaching additional content [188, 153, 207, 131].

5.6 Summary and Future Work

This chapter presents K-12 teaching recommendations for researchers and educators based on the conversational agent concepts presented in Chapter 4; a K-12 AI education literature review [214]; an AI curricula co-design study with K-12 teachers [188]; and studies involving teaching conversational agent development and concepts to K-12 students [99, 185, 193, 191, 215, 190]. Table 5.2 summarizes the teaching recommendations presented in this chapter, as well as those from Chapter 3. These recommendations could be built upon by completing further studies with K-12 teachers in which they use these recommendations in their classroom. Through such studies, researchers could investigate which recommendations are most effective, identify other methods K-12 teachers use as they teach the agent concepts, and generally draw on the teachers' wealth of experience in the classroom.

Table 5.2: Recommendations for teaching K-12 conversational agent concepts and design, and the literature associated with developing these recommendations. The recommendations labeled with a "T" are largely based on the conversational agent concepts themselves, a literature review of K-12 AI education [214] and co-design study with K-12 teachers [188], whereas those labeled with "DR-P" are largely based on the study presented in Chapter 3.

K-12 Teaching Recommendation	**Associated literature**
T1: Incorporate high iterativeness and instant feedback	[214, 99, 215, 191, 218, 209]
T2: Empower students to learn by teaching	[214, 53, 136, 197, 99]
T3: Utilize relevant data	[214, 75, 171, 170, 169, 154, 48, 186, 153, 188]
T4: Utilize gamification and embodiment	[214, 188, 105, 206, 199, 152, 131, 53]
T5: Integrate conversational agent concepts into core curricula	[214, 188, 188, 153, 207, 131, 99]
T6: Reduce the prior experience gap	[214, 191, 83, 50, 65]
DR-P1: Encourage trust of pedagogical agents (to the extent of their trustworthiness) to facilitate learning from them	See Chapter 3
DR-P2: Engage students in activities that will reinforce the concepts being taught with respect to their conversational agent partner models	See Chapter 3
DR-P3: Teach visual programming to empower nearly anyone to program conversational agents	See Chapter 3
DR-P4: Emphasize concepts that are challenging for particular audiences	See Chapter 3
DR-P5: Include both societal impact and conversational agent development activities to facilitate computational action	See Chapter 3
DR-P6: Encourage and provide consistent programming opportunities to underrepresented minorities	See Chapter 3
DR-P7: Supplement conversational agent development activities with additional agent engagement, AI learning and programming activities	See Chapter 3

Chapter 6

Summary of Contributions and Future Work

As conversational agent technology becomes more complex, it becomes simpler to those who use it. When things become simple, we often take them for granted—and yet, these simple things can deeply affect our lives. When we reveal the complexity of simple things, however, we provide people with the opportunity to take part in how they are shaped—and how they shape us. To empower children with this opportunity is to shape the future.

6.1 Main Contributions

For this dissertation, I developed three main conversational agent educational platforms and associated curricula [99, 191, 190]; performed seven user studies, many of which involved these platforms [99, 191, 193, 185, 190, 215, 188]; completed two main systematic literature reviews [214]; and analyzed five conversational agent platforms and associated educational materials [99, 191, 190, 12, 62]. Through this research, I developed the K-12 Conversational Agent Design and Understanding Framework, which includes fourteen conversational agent design recommendations, forty core conversational agent concepts and thirteen K-12 teaching guidelines for conversational agent curricula. In the following sections, I describe my contributions as well as potential areas for future work.

6.1.1 The K-12 Conversational Agent Design and Understanding Framework

In this dissertation, I presented a K-12 Conversational Agent Design and Understanding Framework. Its ultimate goal is to foster accurate understanding and perceptions of agents in K-12 students. The framework is composed of three main components for three main audiences:

1. **Conversational Agent Design Recommendations**
   - The design recommendations are intended for agent developers (K-12 students included) to create transparent, trustworthy agents with high usability, and agents aligned with their audiences' ideal agents.
   - Table 4.2 lists fourteen of the agent design recommendations and Table 3.2 lists three others.
2. **Core Conversational Agent Concepts**
   - The agent concepts are intended for students to develop better understanding of agents. They are also intended to empower researchers and educators to teach students comprehensively about conversational agents.
   - Table 4.8 lists the forty agent concepts.
3. **K-12 Teaching Guidelines**
   - The teaching guidelines are intended for researchers and educators to effectively teach core conversational agent concepts and design guidelines to K-12 students.
   - Table 5.2 lists the thirteen teaching guidelines.

6.1.2 Conversational Agent Platforms I Developed, Performed Studies with, and Analyzed

In this dissertation, I also presented three main educational conversational agent platforms I developed with the support of others in the MIT and AI education community. These platforms allow K-12 students to learn many of the agent concepts through interaction with or development of conversational agents.

1. **Zhorai**
   - Zhorai is a teachable conversational agent for 3-5th grade students.

- Students can learn the concepts listed in Table 4.3 through interacting with Zhorai.
- Tables 4.1 and 4.2 compare the concepts and design recommendations learned through the Zhorai activity with the other activities analyzed in Chapter 4.
- I assessed students' learning through this platform in a study with fourteen 3-5th grade students [99].

2. **ConvoBlocks**
   - ConvoBlocks is a block-based programming interface for 5-12th grade students.
   - Students can learn the concepts listed in Table 4.4 through developing agents using this interface.
   - Tables 4.1 and 4.2 compare the concepts and design recommendations learned through the ConvoBlocks activity with the other activities analyzed in Chapter 4.
   - I assessed students' learning and their perceptions of conversational agents when using this platform in multiple studies, with over 150 participants total [185, 191, 193].
3. **Convo**
   - Convo is a natural language programming agent for 7-8th grade students.
   - Students can learn the concepts listed in Table 4.5 through developing agents using this interface.
   - Tables 4.1 and 4.2 compare the concepts and design recommendations learned through the Convo activity with the other activities analyzed in Chapter 4.
   - I assessed students' learning when using this platform and Convo's usability in multiple studies, with 60 participants total [190, 215].

6.1.3 Industry Conversational Agent Platforms I Analyzed

I identified core conversational agent concepts through analyzing the platforms in the previous section, as well as two other prominent industry conversational agent development platforms, and their associated educational materials. The industry platforms included:

1. **The Alexa Developer Console**
   - The Alexa Developer Console is Amazon's conversational agent development platform for software developers [12].

- Developers can learn the concepts listed in Table 4.6 through creating agents using this interface.
- Tables 4.1 and 4.2 compare the concepts and design recommendations learned through the Alexa Developer Console activity with the other activities analyzed in Chapter 4.

2. **Dialogflow CX**
   - Dialogflow CX is Google's conversational agent development platform for software developers [62].
   - Developers can learn the concepts listed in Table 4.7 through creating agents using this interface.
   - Tables 4.1 and 4.2 compare the concepts and design recommendations learned through the Dialogflow activity with the other activities analyzed in Chapter 4.

6.1.4 Systematic Literature Reviews

I also presented two systematic literature reviews, which informed the Conversational Agent Design Recommendations as well as the K-12 Teaching Guidelines:

1. **Conversational Agent Usability Design Guidelines Literature Review**
   - To identify prominent, comprehensive and current conversational agent usability design guidelines, I performed a snowball sampling literature review.
   - The review identified 82 publications (as noted in Table 2.1), 16 of which fell entirely under the criteria in Table 2.2.
   - The final, most prominent paper with aggregated usability guidelines identified was Murad et al.'s article, "Design guidelines for hands-free speech interaction" [117], which surveyed 21 papers to develop its 12 usability guidelines for speech interfaces.
   - Table 2.3 shows the final 12 guidelines.
2. **K-12 AI Education Literature Review [214]**
   - To identify teaching methods and future research opportunities related to AI educational tools, I performed a snowball sampling literature review alongside we Zhou et al. [214].
   - The review identified 49 publications that fell under the required criteria.

- Through analyzing the 49 publications, we identified nine future opportunities for K-12 AI education, as well as eleven teaching guidelines for K-12 AI education.
- Through analyzing the eleven AI teaching guidelines, I identified six that were particularly applicable to teaching conversational AI (and that were unique with respect to the other seven conversational agent teaching guidelines developed in Chapter 3), and discussed their relevance to the major conversational agent concept categories.
- Table 5.2 shows the final six teaching guidelines from the literature review, and Table 5.1 shows their relevance to the concept categories.

6.2 Future Work

There are many opportunities to further develop the K-12 Conversational Agent Design and Understanding Framework. For instance, additional concepts could be added as the conversational agent landscape grows and develops. Additional studies could investigate the differences in how students of different age levels learn and understand the conversational agent concepts, and how teachers at different schools (e.g., schools in different countries, schools with different socioeconomic statuses, etc.) can most effectively teach the concepts.

Researchers could also further investigate how students' perceptions change. For example, in studies similar to the one presented in Chapter 3, researchers could investigate changes with respect to various agents (e.g., using the Google Assistant vs. Alexa) or in various contexts (e.g., in an after-school program vs. in a regular K-12 school classroom). They could also investigate how perceptions change if students only learned concepts from a certain subset of categories (e.g., only from Natural Language Understanding or Dialog Management) to determine the most vital concepts for students to learn.

Another important research question has to do with how students respond to information provided by agents depending on their trust of the agents. For instance, if students interacted with an agent that only provided correct information some of the time, how often would they act on the information provided? For example, if an agent told students, "It's good to eat jellybeans while writing a test", how many students would eat jellybeans?

Researchers performed a similar study in the medical field and found clinician's reliance on agents' information did not always correspond to their reported level of trust of the information [59]. By investigating students' trust of and reliance on agents, we can better understand what healthy levels of trust are towards different agents.

Through further investigation into students' perceptions, observation of students' responses to agents, and identification of agent concepts as the field changes and grows, researchers can continue to develop the K-12 Conversational Agent Design and Understanding Framework. With this framework, educators, researchers and agent developers can better prepare K-12 students for an agent-filled world.

Appendix A

Review of AI Education Resources

This appendix contains tables originally from the literature review I completed alongside Zhou and Lin [214]. It outlines the AI competencies addressed by 49 different AI education resources. Table A.1 lists the competencies, which are originally from a paper by Long and Magerko [100]. Table A.2 lists teaching recommendations for AI educators, which are from the same source [100].

Table A.1: List of AI Competencies from [100].

#	Competency	#	Competency
1	Recognizing AI	10	Human Role in AI
2	Understanding Intelligence	11	Data Literacy
3	Interdisciplinarity	12	Learning from Data
4	General vs. Narrow	13	Critically Interpreting Data
5	AI Strengths & Weaknesses		
6	Imagine Future AI	14	Action & Reaction
7	Representations	15	Sensors
8	Decision-Making	16	Ethics
9	Machine Learning Steps	17	Programmability

Table A.2: List of AI Teaching Recommendations from [100].

#	Consideration	#	Consideration
1	Explainability	9	Identity, Values, & Backgrounds
2	Embodied Interactions	10	Support for Parents
3	Contextualizing Data	11	Social Interaction
4	Promote Transparency	12	Leverage Learners' Interests
5	Unveil Gradually	13	Acknowledging Preconceptions
6	Opportunities to Program	14	New Perspectives
7	Milestones	15	Low Barrier to Entry
8	Critical Thinking		

	Competencies																						
	What is AI?				What can AI do?		How does AI work?									How should AI be used? & programmability							
Key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Targ.	Time	Tool	Scfld.	Ceil.	Teach.
One-Year [41]	✓	✓		✓		✓	✓	✓	✓	✓		✓	✓		✓	✓	✓	M	Y	W	I	H	T:D
Robot [66]	✓	✓								✓				✓		✓	✓	M	W	P	I	H	R:I
Cognimate [9]	✓					✓	✓	✓	✓	✓					✓		✓	P	N	W	I	H	R:I
Alternate [7]	✓					✓		✓	✓			✓			✓	✓		H	W	W	I	L	T:I
Decoding [8]	✓					✓			✓			✓				✓		M	H	U	I	L	N
For-All [63]	✓						✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	M	M	W	I	H	T:I
K-Uni [24]	✓						✓	✓	✓		✓	✓						A	W	P	I	H	R:I
Base [44]	✓							✓	✓	✓	✓	✓					✓	P	N	W	T	L	N
Co-design [53]	✓							✓	✓	✓		✓	✓				✓	P	D	W	I	H	R:I
IRobot [6]		✓	✓				✓	✓						✓			✓	H	W	P	I	H	R:I
Classroom [19]		✓	✓					✓	✓	✓	✓	✓					✓	M	W	P	I	L	T:I
Conver [59]		✓			✓	✓	✓	✓		✓		✓				✓	✓	M/H	W	W	T	H	R:I
Why-Not [15]		✓			✓		✓	✓	✓	✓	✓	✓	✓			✓		H	D	W	T	L	N
Game [37]		✓						✓	✓			✓						M/H	H	U	I	N	T:L
Popbots [68]			✓				✓	✓	✓	✓		✓		✓			✓	P	W	P	I	L	N
Smiley [64]					✓		✓	✓	✓	✓	✓	✓						H	H	W	T	L	N
Zhorai [29]					✓		✓	✓	✓	✓		✓	✓			✓		P	H	W	T/I	L	R:I
STEM [42]						✓	✓	✓	✓	✓	✓	✓	✓				✓	M	D	W	I	H	T:I
App-Inv [34]						✓	✓	✓	✓	✓		✓	✓		✓	✓	✓	M/H	W	W	I	H	N
Wolfram [71]							✓	✓	✓	✓	✓	✓						A	N	W	I	H	N
Summer [35]							✓	✓	✓	✓	✓	✓						H	W	W	I	H	T:D

[1]Targ.: P/M/H/U/A/N - Primary/Middle/High/Undergrad/All/Not specified;
Time: H/D/M/Y/N - measured in Hours/Days/Months/Years/Not specified;
Tool requirement: P/W/U - Physical(tangible)/Web(software)/Unplugged (Note: Physical trumped Web trumped Unplugged);
Scaffolding: I/T/N - Instructional scaffolding / Tool scaffolding / Not specified;
Ceilingness: H/L - High/Low ceiling;
Teacher involvement: R:I/T:I/T:L/T:D/N - Researchers as Instructors / Teachers as Instructors/ Teachers as Learners / Teachers as Designers / Not specified

	Competencies																						
	What is AI?				What can AI do?		How does AI work?									How should AI be used? & programmability							
Key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Targ.	Time	Tool	Scfld.	Ceil.	Teach.
Tensorflow [45]							✓	✓	✓	✓		✓	✓					H/U	N	W	N	H	N
Sports [76]							✓	✓	✓	✓		✓	✓					M/H	H	P	I	H	R:I
Gentle [14]							✓	✓	✓	✓		✓					✓	H	W	W	I	H	T:I
High-Sch [31]							✓	✓	✓	✓		✓						P/H	H	W	I	H	T:I
Youth [75]							✓	✓	✓			✓	✓		✓		✓	P/M	D	P	I	H	R:I
Nodes [2]							✓	✓	✓			✓	✓				✓	N	N	P	N	H	N
Control [52]							✓	✓		✓	✓				✓		✓	U	N	W	N	H	N
Bots [36]							✓	✓		✓		✓	✓	✓				N	N	P	N	L	N
Cozmo [55]							✓	✓		✓				✓	✓		✓	A	N	P	I	H	N
Logic [73]							✓	✓		✓								M	W	W	I	H	T:D
Semantic[25]							✓	✓										P	H	W	T	L	N
Secondary [13]							✓		✓	✓		✓	✓					H	N	W	N	H	T:I
Grinch [60]							✓			✓	✓	✓				✓	✓	N	N	W	N	H	N
Gender [57]							✓			✓	✓		✓					H	W	W	I	H	T:I
Boxes [18]								✓	✓	✓		✓	✓			✓		M	N	P	I	L	N
Image [50]								✓	✓	✓		✓	✓				✓	H	D	W	I	H	T:I
Apps [74]								✓	✓	✓		✓	✓				✓	H/U	W	W	I	H	N
ML4Kids [26]								✓	✓	✓		✓	✓				✓	N	N	W	N	H	N
Whom [61]								✓	✓	✓		✓	✓				✓	P	H	W	I	H	R:I
Snap! [22]									✓	✓		✓			✓		✓	H	N	W	N	H	N
Cloud [21]										✓		✓			✓		✓	M/U	N	W	N	H	N
Multiyear [17]										✓					✓		✓	P	Y	P	I	L	T:I
Data-Kids [48]											✓					✓		M	H	W	I	L	T:I
Any-Cubes [43]												✓						N	N	P	N	H	N
Sensors [28]															✓			P	H	P	I	H	T:I
Cards [5]																✓		H	W	U	I	H	T:I
Software [47]																		H	M	W	N	H	T:I

Appendix B

Surveys from the Conversational Agent Development and Perceptions Study

This appendix contains the three surveys provided to participants in the Conversational Agent Development and Perceptions Study discussed in Chapter 3. The first survey was provided prior to the entire workshop, the second was provided after the programming activity, and the third was provided after the entire workshop.

Before you start...

🌍 We're excited to have you join Future World Challenge – MIT Edition, which is a global intergenerational research study on the topic of conversational Artificial Intelligence (AI).

🔎 As a reminder, the goal of the study is to investigate how people perceive conversational agents, like Alexa or Siri. Alexa is a voice assistant created by Amazon that can have conversations through speaking and listening to humans. It can set reminders, tell you the weather, and play music, among other things.

We would like to understand your *initial* thoughts about Alexa (even if you haven't interacted with Alexa before), so 👉 please don't look Alexa up online or ask anyone about Alexa 👈. If you do not know what Alexa is or have never experienced interactions with Alexa, please base your answers on other voice assistants, like the Google Assistant or Siri, or consider putting down "neutral" answers.

* Required

1. I am... *

   Mark only one oval.

   ◯ 11-18 years old
   ◯ A parent or legal guardian of a child 11-18 years old
   ◯ A grandparent of a child 11-18 years old

2. I agree to NOT discuss my answers with anyone (including my child/parent/grandparent) until after the entire Future Worlds Challenge has completed *

   Mark only one oval.

   ◯ Yes

3. I have interacted with these conversational agent(s) before: *

Check all that apply.

- [ ] Amazon Alexa
- [ ] Google Assistant
- [ ] Apple's Siri
- [ ] None of the above (or other agents) to my knowledge

Other: [ ] ______________________

4. I have programmed before using... *

Check all that apply.

- [ ] A visual or blocks-based interface, like Scratch or MIT App Inventor
- [ ] A text-based language, like Python or JavaScript
- [ ] None of the above (or other programming languages) to my knowledge

Other: [ ] ______________________

5. I have learned about AI before in a workshop or lesson. *

Mark only one oval.

( ) Yes
( ) No
( ) Other: ______________________

Perception of Alexa

Don't worry if you haven't interacted with Alexa before: If you haven't, please base your answers on other voice assistants, like the Google Assistant or Siri, or consider putting down "neutral" answers.

6. Do you think Alexa is more machine-like or human-like? *

Mark only one oval.

	1	2	3	4	5	
Machine-like	◯	◯	◯	◯	◯	Human-like

7. Do you think Alexa is more inflexible (like someone who has a hard time changing plans) or flexible (like someone who can easily change plans)? *

Mark only one oval.

	1	2	3	4	5	
Inflexible	◯	◯	◯	◯	◯	Flexible

8. Do you think Alexa is more of a friend or a co-worker (in this case, someone you might work on a project with, but wouldn't hang out with)? *

Mark only one oval.

	1	2	3	4	5	
Friend	◯	◯	◯	◯	◯	Co-worker

9. Do you think Alexa is more warm (like someone who is friendly/caring) or cold (like someone who is unfriendly/uncaring)? *

Mark only one oval.

	1	2	3	4	5	
Warm	◯	◯	◯	◯	◯	Cold

10. Do you think Alexa is more unreliable (like someone you wouldn't trust) or dependable (like someone you can trust)? *

Mark only one oval.

	1	2	3	4	5	
Unreliable	◯	◯	◯	◯	◯	Dependable

11. Do you think Alexa is more interactive (like someone who listens and talks back and forth) or start-stop (like someone who doesn't continue the conversation)? *

Mark only one oval.

	1	2	3	4	5	
Interactive	◯	◯	◯	◯	◯	Start-stop

12. Do you think Alexa is more competent (like someone who can understand and do things well) or incompetent (like someone who has a hard time understanding or doesn't do a good job)? *

Mark only one oval.

	1	2	3	4	5	
Competent	◯	◯	◯	◯	◯	Incompetent

13. Do you think Alexa is more like an authority figure (like a teacher) or peer (like a classmate)? *

Mark only one oval.

	1	2	3	4	5	
Authority figure	◯	◯	◯	◯	◯	Peer

Knowledge-sharing

14. I see myself as a computer programmer *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

15. I am confident I can design and create my own technology project, not just something someone tells me to create. *

Mark only one oval.

	1	2	3	4	5	
Strong disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

16. I am confident I can make an impact in my community or in the world using technology. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

17. I trust conversational agents (e.g., Siri, Alexa, Google Home) *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

18. Conversational agents (e.g., Siri, Alexa, Google Home) say things that are... *

Mark only one oval.

	1	2	3	4	5	
Always right	◯	◯	◯	◯	◯	Always wrong

19. Please explain why you think conversational agents say things that are right/wrong. *

20. If you asked the people/systems below questions about recent events (e.g., things in the news), how often do you think their responses would be correct vs. incorrect? *

Mark only one oval per row.

	Always incorrect	Incorrect most of the time	About equal amounts incorrect and correct	Correct most of the time	Always correct
If I asked my parents about the news	◯	◯	◯	◯	◯
If I looked in the newspaper	◯	◯	◯	◯	◯
If I asked Alexa about the news	◯	◯	◯	◯	◯
If I asked my friend about the news	◯	◯	◯	◯	◯
If I did a Google search for news	◯	◯	◯	◯	◯

Demographics

21. When I use conversational devices (Alexa, Siri, Google Home, etc.), I generally use them in the language I know best. (E.g., if you feel most comfortable speaking German, but you use Alexa in English, answer "No"). *

Mark only one oval.

- ◯ Yes, I generally use conversational agents in the language I know best.
- ◯ No, I generally use conversational agents in another language.
- ◯ Other: ____________________

22. What is your mother language? (Or the language you feel you know best?) *

23. How old are you? *

24. What is your gender identity? *

25. Which country do you consider your "home"? (E.g., the place where you've lived for most of your life.) *

26. What is your code name? (Note: This should have been emailed to you when you registered. If you don't know it, ask one of the leaders, like Jessica, in the Zoom-chat.) *

Future Worlds Challenge: Mid-Survey

As a reminder, 👉 please don't look Alexa up online or ask anyone about Alexa and please don't talk to anyone about your answers until the entire Future Worlds Challenge is done 👈.

* Required

1. What is your code name? (Note: This should have been emailed to you when you registered. If you don't know it, ask one of the leaders, like Jessica, in the Zoom-chat.) *

   ____

2. I am... *

   Mark only one oval.

   - ⬭ 11-18 years old
   - ⬭ A parent or legal guardian of a child 11-18 years old
   - ⬭ A grandparent of a child 11-18 years old

3. I agree to NOT discuss my answers with anyone (including my child/parent/grandparent) until after the entire Future Worlds Challenge has completed *

   Mark only one oval.

   - ⬭ Yes

Perception of Alexa

1/7

4. Do you think Alexa is more machine-like or human-like? *

Mark only one oval.

	1	2	3	4	5	
Machine-like	◯	◯	◯	◯	◯	Human-like

5. Do you think Alexa is more inflexible (like someone who has a hard time changing plans) or flexible (like someone who can easily change plans)? *

Mark only one oval.

	1	2	3	4	5	
Inflexible	◯	◯	◯	◯	◯	Flexible

6. Do you think Alexa is more of a friend or a co-worker (in this case, someone you might work on a project with, but wouldn't hang out with)? *

Mark only one oval.

	1	2	3	4	5	
Friend	◯	◯	◯	◯	◯	Co-worker

7. Do you think Alexa is more warm (like someone who is friendly/caring) or cold (like someone who is unfriendly/uncaring)? *

Mark only one oval.

	1	2	3	4	5	
Warm	◯	◯	◯	◯	◯	Cold

8. Do you think Alexa is more unreliable (like someone you wouldn't trust) or dependable (like someone you can trust)? *

Mark only one oval.

	1	2	3	4	5	
Unreliable	◯	◯	◯	◯	◯	Dependable

9. Do you think Alexa is more interactive (like someone who listens and talks back and forth) or start-stop (like someone who doesn't continue the conversation)? *

Mark only one oval.

	1	2	3	4	5	
Interactive	◯	◯	◯	◯	◯	Start-stop

10. Do you think Alexa is more competent (like someone who can understand and do things well) or incompetent (like someone who has a hard time understanding or doesn't do a good job)? *

Mark only one oval.

	1	2	3	4	5	
Competent	◯	◯	◯	◯	◯	Incompetent

11. Do you think Alexa is more like an authority figure (like a teacher) or peer (like a classmate)? *

Mark only one oval.

	1	2	3	4	5	
Authority figure	◯	◯	◯	◯	◯	Peer

Knowledge-sharing

12. I see myself as a computer programmer *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

13. I feel like I can design and create my own technology project, not just something someone tells me to create. *

Mark only one oval.

	1	2	3	4	5	
Strong disagree	◯	◯	◯	◯	◯	Strongly agree

14. I feel like I can make an impact in the world using technology. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

15. I trust conversational agents (e.g., Siri, Alexa, Google Home) *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

16. Conversational agents (e.g., Siri, Alexa, Google Home) say things that are... *

Mark only one oval.

	1	2	3	4	5	
Always right	◯	◯	◯	◯	◯	Always wrong

17. Please explain why you think conversational agents say things that are right/wrong. *

18. If you asked the people/systems below questions about recent events (e.g., things in the news), how often do you think their responses would be correct vs. incorrect? *

Mark only one oval per row.

	Always incorrect	Incorrect most of the time	About equal amounts incorrect and correct	Correct most of the time	Always correct
If I asked my parents about the news					
If I looked in the newspaper					
If I asked Alexa about the news					
If I asked my friend about the news					
If I did a Google search for news					

19. What were the most difficult concepts for you to learn about conversational agents? (You can choose multiple.) Learning about... *

Check all that apply.

- [ ] How agents need to be trained with examples or "utterances"
- [ ] How you can say different things (an "intent") to an agent and it still understands
- [ ] How you need to say the name of the Alexa skill ("invocation name") for it to understand
- [ ] How to make Alexa get (or use) specific information, like a number, from the user (i.e., using "slots")
- [ ] How to make Alexa do or say things (making the "endpoint function")
- [ ] The terminology (e.g., what "invocation name" or "intent" means)
- [ ] How to type/say something so that Alexa understands ("testing")
- [ ] How to get Alexa to ask a follow-up question (the "ask" block)

Other: [ ] ____________________

20. Any other thoughts/comments on what was challenging? *

7/7

Future Worlds Challenge: Final Survey

As a reminder, 👉 please don't look Alexa up online or ask anyone about Alexa and please don't talk to anyone about your answers until the entire Future Worlds Challenge is done 👈.

* Required

1. What is your code name? (Note: This should have been emailed to you when you registered. If you don't know it, ask one of the leaders, like Jessica, in the Zoom-chat.) *

   ____

2. I am... *

   Mark only one oval.

   ⬭ 11-18 years old
   ⬭ A parent or legal guardian of a child 11-18 years old
   ⬭ A grandparent of a child 11-18 years old

3. I agree to NOT discuss my answers with anyone (including my child/parent/grandparent) until after the entire Future Worlds Challenge has completed *

   Mark only one oval.

   ⬭ Yes

Perception of Alexa

4. Do you think Alexa is more machine-like or human-like? *

Mark only one oval.

	1	2	3	4	5	
Machine-like	◯	◯	◯	◯	◯	Human-like

5. Do you think Alexa is more inflexible (like someone who has a hard time changing plans) or flexible (like someone who can easily change plans)? *

Mark only one oval.

	1	2	3	4	5	
Inflexible	◯	◯	◯	◯	◯	Flexible

6. Do you think Alexa is more of a friend or a co-worker (in this case, someone you might work on a project with, but wouldn't hang out with)? *

Mark only one oval.

	1	2	3	4	5	
Friend	◯	◯	◯	◯	◯	Co-worker

7. Do you think Alexa is more warm (like someone who is friendly/caring) or cold (like someone who is unfriendly/uncaring)? *

Mark only one oval.

	1	2	3	4	5	
Warm	◯	◯	◯	◯	◯	Cold

8. Do you think Alexa is more unreliable (like someone you wouldn't trust) or dependable (like someone you can trust)? *

Mark only one oval.

	1	2	3	4	5	
Unreliable						Dependable

9. Do you think Alexa is more interactive (like someone who listens and talks back and forth) or start-stop (like someone who doesn't continue the conversation)? *

Mark only one oval.

	1	2	3	4	5	
Interactive						Start-stop

10. Do you think Alexa is more competent (like someone who can understand and do things well) or incompetent (like someone who has a hard time understanding or doesn't do a good job)? *

Mark only one oval.

	1	2	3	4	5	
Competent						Incompetent

11. Do you think Alexa is more like an authority figure (like a teacher) or peer (like a classmate)? *

Mark only one oval.

	1	2	3	4	5	
Authority figure						Peer

3/7

12. Do you think you changed your opinions on any of the above questions since going through the activities? If so, what changed and why do you think they changed? If not, why do you think they stayed the same? (At least 1-2 sentences) *

Knowledge-sharing

13. I see myself as a computer programmer *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

14. I am confident I can design and create my own technology project, not just something someone tells me to create. *

Mark only one oval.

	1	2	3	4	5	
Strong disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

15. I am confident I can make an impact in my community or in the world using technology. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

16. I trust conversational agents (e.g., Siri, Alexa, Google Home) *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	⬭	⬭	⬭	⬭	⬭	Strongly agree

17. Conversational agents (e.g., Siri, Alexa, Google Home) say things that are... *

Mark only one oval.

	1	2	3	4	5	
Always right	⬭	⬭	⬭	⬭	⬭	Always wrong

18. Do you think you changed your opinion on whether conversational agents say things that are right/wrong since going through the activities? If so, how did it change and why do you think it changed? If not, why do you think it stayed the same? (At least 1-2 sentences) *

____

____

____

____

____

19. If you asked the people/systems below questions about recent events (e.g., things in the news), how often do you think their responses would be correct vs. incorrect? *

Mark only one oval per row.

	Always incorrect	Incorrect most of the time	About equal amounts incorrect and correct	Correct most of the time	Always correct
If I asked my parents about the news	⬭	⬭	⬭	⬭	⬭
If I looked in the newspaper	⬭	⬭	⬭	⬭	⬭
If I asked Alexa about the news	⬭	⬭	⬭	⬭	⬭
If I asked my friend about the news	⬭	⬭	⬭	⬭	⬭
If I did a Google search for news	⬭	⬭	⬭	⬭	⬭

20. Did you finish building the two tutorials and get the Alexa Skills working yesterday? *

Mark only one oval.

- ⬭ I finished both tutorials and did the third tutorial on my own too
- ⬭ I finished both tutorials
- ⬭ I only finished the first tutorial
- ⬭ I only finished the second tutorial
- ⬭ Other: ____________________

21. What did you like most about the workshops? *

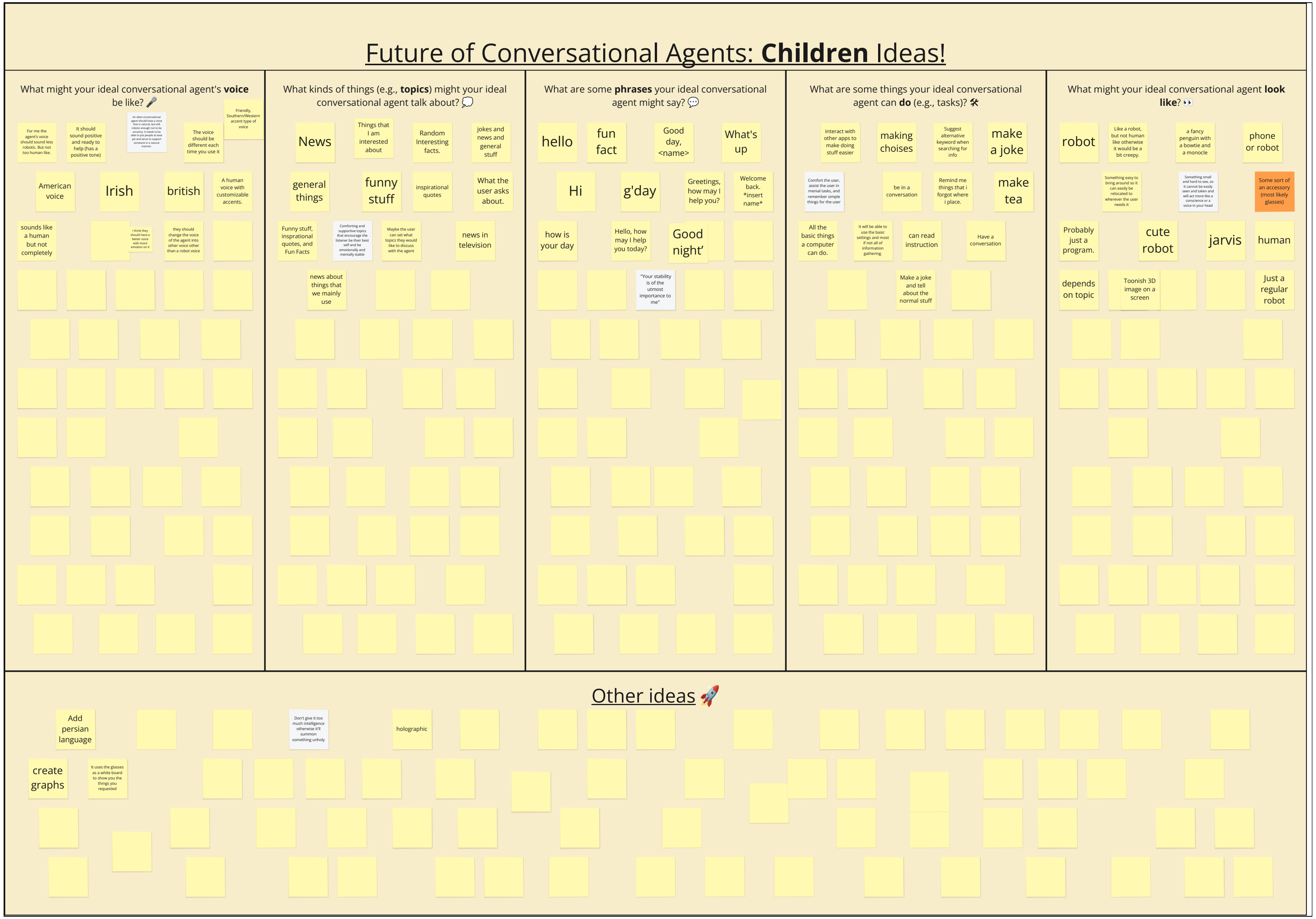
22. What could be improved? *

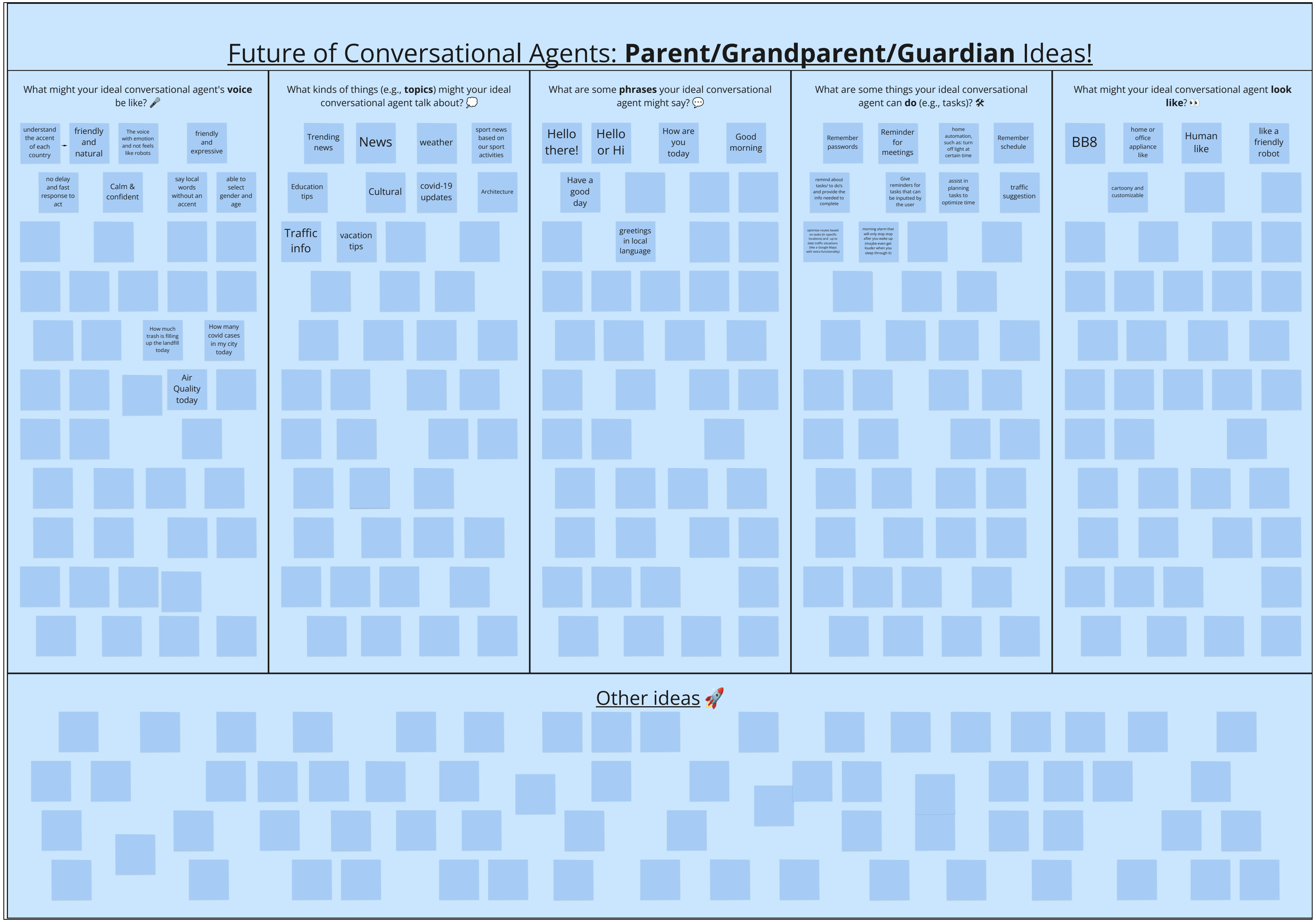
7/7

Appendix C

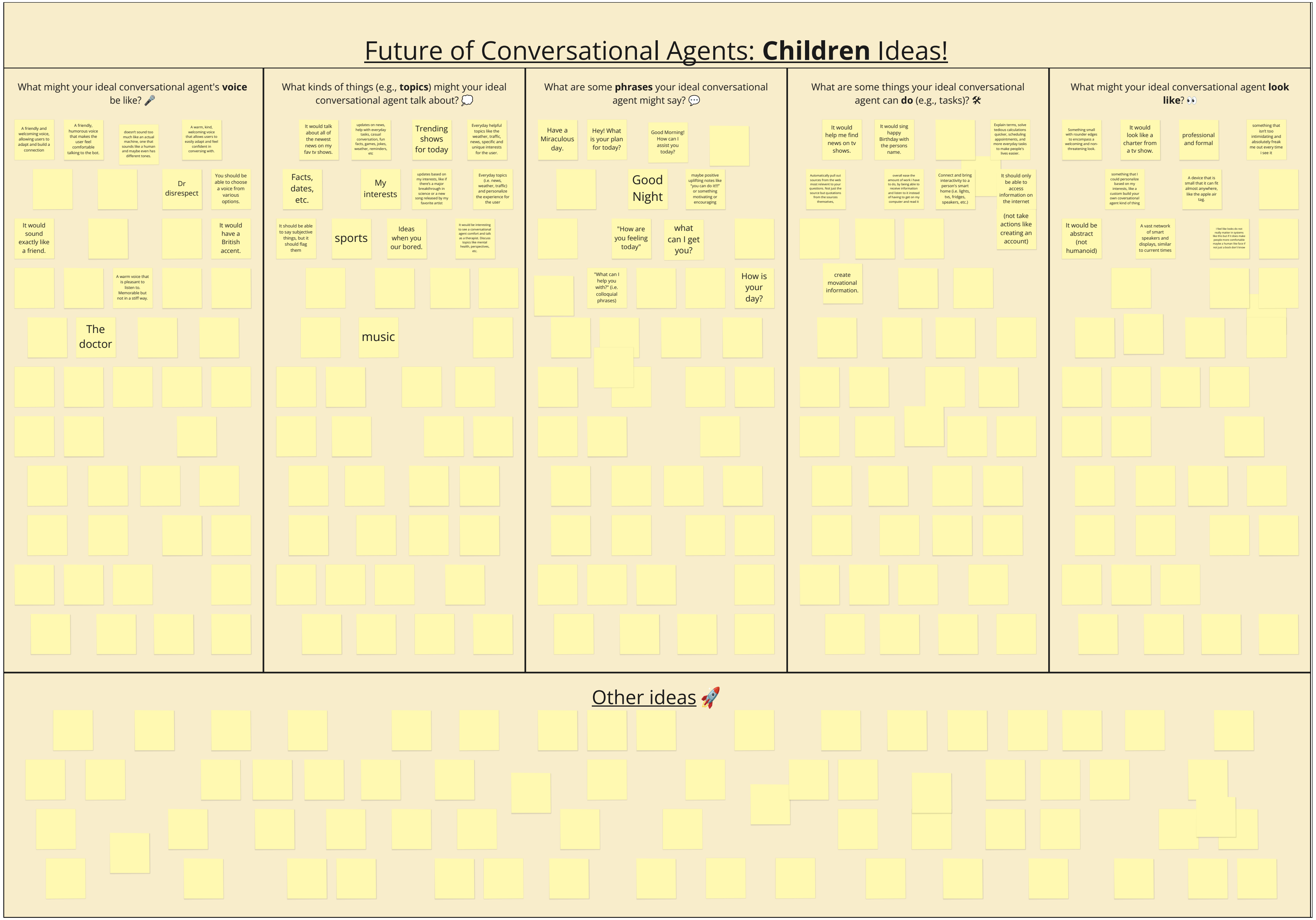
Ideation Boards from the Conversational Agent Development and Perceptions Study

This appendix contains the ideation boards from the study in Chapter 3. As a part of the study, participants came up with ideas for future conversational agents and put virtual sticky-notes on shared Miro boards [110]. This activity was completed by parents and children from both of the workshops (WEIRD and non-WEIRD). The following pages show the participants' final boards in the following order: (1) children from non-WEIRD countries' board, (2) parents from non-WEIRD countries' board, (3) children from WEIRD countries' board, and (4) parents from WEIRD countries board.

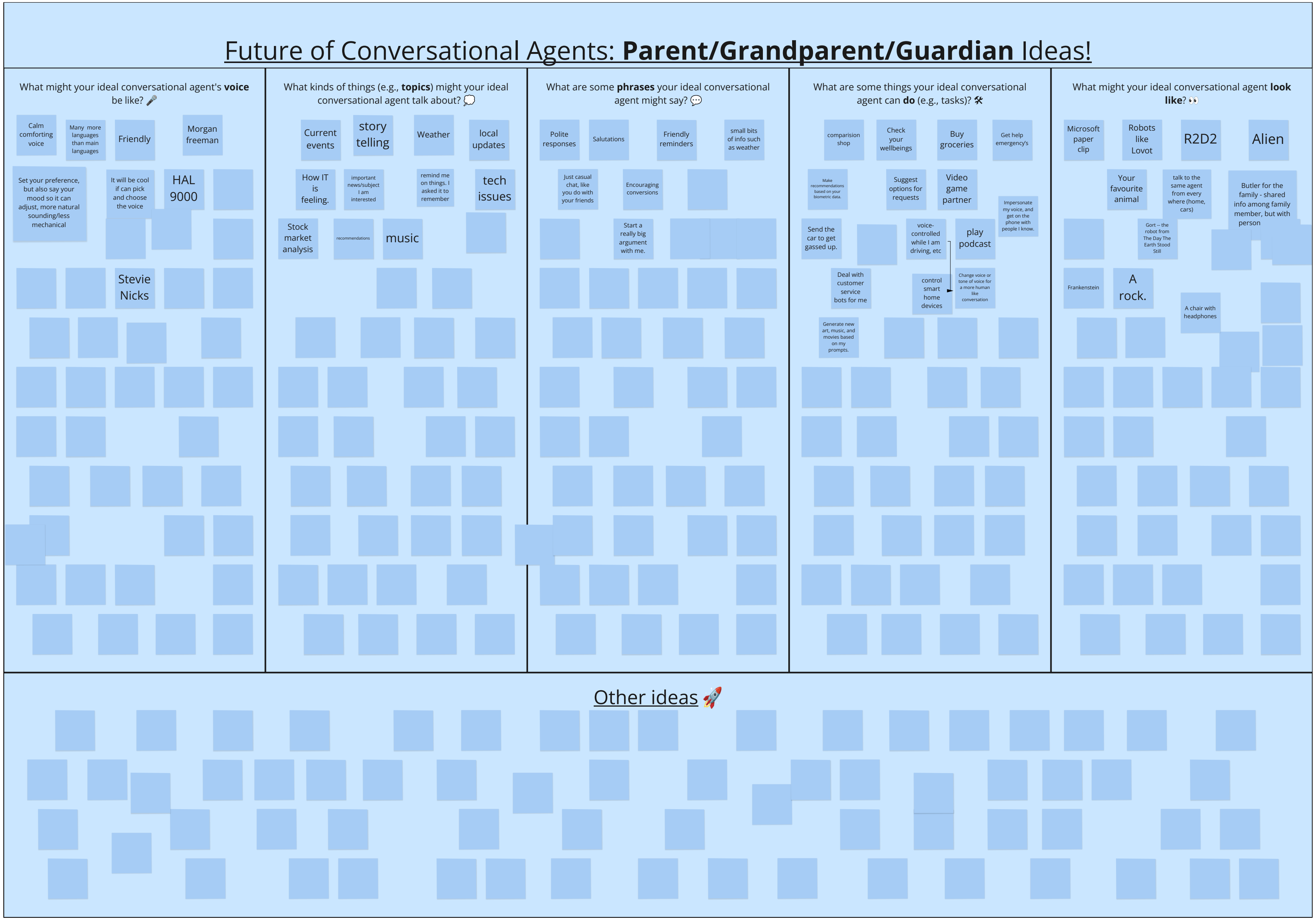

Future of Conversational Agents: Children Ideas!
What might your ideal conversational agent's voice be like?
For me the agent's voice should sound less robotic. But not too human like.
It should sound positive and ready to help (has a positive tone)
An ideal conversational agent should have a voice that is natural, but still robotic enough not to be uncanny. It needs to be able to put people at ease yet and serve to support someone in a natural manner.
The voice should be different each time you use it
Friendly, Southern/Western accent type of voice
American voice
Irish
british
A human voice with customizable accents.
sounds like a human but not completely
I think they should have a better voice with more emotion on it
they should change the voice of the agent into other voice other than a robot voice
What kinds of things (e.g., topics) might your ideal conversational agent talk about?
News
Things that I am interested about
Random Interesting facts.
jokes and news and general stuff
general things
funny stuff
inspirational quotes
What the user asks about.
Funny stuff, inspirational quotes, and Fun Facts
Comforting and supportive topics that encourage the listener be their best self and be emotionally and mentally stable
Maybe the user can set what topics they would like to discuss with the agent
news in television
news about things that we mainly use
What are some phrases your ideal conversational agent might say?
hello
fun fact
Good day, <name>
What's up
Hi
g'day
Greetings, how may I help you?
Welcome back. *insert name*
how is your day
Hello, how may I help you today?
Good night’
"Your stability is of the utmost importance to me"
What are some things your ideal conversational agent can do (e.g., tasks)?
interact with other apps to make doing stuff easier
making choises
Suggest alternative keyword when searching for info
make a joke
Comfort the user, assist the user in menial tasks, and remember simple things for the user
be in a conversation
Remind me things that i forgot where i place.
make tea
All the basic things a computer can do.
It will be able to use the basic settings and most if not all of information gathering.
can read instruction
Have a conversation
Make a joke and tell about the normal stuff
What might your ideal conversational agent look like?
robot
Like a robot, but not human like otherwise it would be a bit creepy.
a fancy penguin with a bowtie and a monocle
phone or robot
Something easy to bring around so it can easily be relocated to wherever the user needs it
Something small and hard to see, so it cannot be easily seen and taken and will act more like a conscience or a voice in your head
Some sort of an accessory (most likely glasses)
Probably just a program.
cute robot
jarvis
human
depends on topic
Toonish 3D image on a screen
Just a regular robot
Other ideas
Add persian language
Don't give it too much intelligence otherwise it'll summon something unholy
holographic
create graphs
It uses the glasses as a white board to show you the things you requested



Future of Conversational Agents: Parent/Grandparent/Guardian Ideas!
What might your ideal conversational agent's voice be like? 🎤
understand the accent of each country
friendly and natural
The voice with emotion and not feels like robots
friendly and expressive
no delay and fast response to act
Calm & confident
say local words without an accent
able to select gender and age
How much trash is filling up the landfill today
How many covid cases in my city today
Air Quality today
What kinds of things (e.g., topics) might your ideal conversational agent talk about? 💭
Trending news
News
weather
sport news based on our sport activities
Education tips
Cultural
covid-19 updates
Architecture
Traffic info
vacation tips
What are some phrases your ideal conversational agent might say? 💬
Hello there!
Hello or Hi
How are you today
Good morning
Have a good day
greetings in local language
What are some things your ideal conversational agent can do (e.g., tasks)? 🛠
Remember passwords
Reminder for meetings
home automation, such as: turn off light at certain time
Remember schedule
remind about tasks/ to do's and provide the info needed to complete
Give reminders for tasks that can be inputted by the user
assist in planning tasks to optimize time
traffic suggestion
optimize routes based on tasks (in specific locations) and up to date traffic situations (like a Google Maps with extra functionality)
morning alarm that will only stop stop after you wake up (maybe even get louder when you sleep through it)
What might your ideal conversational agent look like? 👀
BB8
home or office appliance like
Human like
like a friendly robot
cartoony and customizable
Other ideas 🚀



Future of Conversational Agents: Children Ideas!
What might your ideal conversational agent's voice be like? 🎤
A friendly and welcoming voice, allowing users to adapt and build a connection
A friendly, humorous voice that makes the user feel comfortable talking to the bot.
doesn't sound too much like an actual machine, one that sounds like a human and maybe even has different tones.
A warm, kind, welcoming voice that allows users to easily adapt and feel confident in conversing with.
Dr disrespect
You should be able to choose a voice from various options.
It would sound exactly like a friend.
It would have a British accent.
A warm voice that is pleasant to listen to. Memorable but not in a stiff way.
The doctor
What kinds of things (e.g., topics) might your ideal conversational agent talk about? 💭
It would talk about all of the newest news on my fav tv shows.
updates on news, help with everyday tasks, casual conversation, fun facts, games, jokes, weather, reminders, etc
Trending shows for today
Everyday helpful topics like the weather, traffic, news, specific and unique interests for the user.
Facts, dates, etc.
My interests
updates based on my interests, like if there's a major breakthrough in science or a new song released by my favorite artist
Everyday topics (i.e. news, weather, traffic) and personalize the experience for the user
It should be able to say subjective things, but it should flag them
sports
Ideas when you our bored.
It would be interesting to see a conversational agent comfort and talk as a therapist. Discuss topics like mental health, perspectives, etc.
music
What are some phrases your ideal conversational agent might say? 💬
Have a Miraculous day.
Hey! What is your plan for today?
Good Morning! How can I assist you today?
Good Night
maybe positive uplifting notes like "you can do it!!!" or something motivating or encouraging
"How are you feeling today"
what can I get you?
"What can I help you with?" (i.e. colloquial phrases)
How is your day?
What are some things your ideal conversational agent can do (e.g., tasks)? 🛠
It would help me find news on tv shows.
It would sing happy Birthday with the persons name.
Explain terms, solve tedious calculations quicker, scheduling appointments, and more everyday tasks to make people's lives easier.
Automatically pull out sources from the web most relevent to your questions. Not just the source but quotations from the sources themselves,
overall ease the amount of work i have to do, by being able to receive information and listen to it instead of having to get on my computer and read it
Connect and bring interactivity to a person's smart home (i.e. lights, tvs, fridges, speakers, etc.)
It should only be able to access information on the internet
(not take actions like creating an account)
create movational information.
What might your ideal conversational agent look like? 👀
Something small with rounder edges to encompass a welcoming and non-threatening look.
It would look like a charter from a tv show.
professional and formal
something that isn't too intimidating and absolutely freak me out every time i see it
something that I could personalize based on my interests, like a custom build your own coversational agent kind of thing
A device that is small that it can fit almost anywhere, like the apple air tag.
It would be abstract (not humanoid)
A vast network of smart speakers and displays, similar to current times
I feel like looks do not really matter in systems like this but if it does make people more comfortable maybe a human like face if not just a bock don't know
Other ideas 🚀


Future of Conversational Agents: Parent/Grandparent/Guardian Ideas!
What might your ideal conversational agent's voice be like? 🎤
Calm comforting voice
Many more languages than main languages
Friendly
Morgan freeman
Set your preference, but also say your mood so it can adjust, more natural sounding/less mechanical
It will be cool if can pick and choose the voice
HAL 9000
Stevie Nicks
What kinds of things (e.g., topics) might your ideal conversational agent talk about? 💭
Current events
story telling
Weather
local updates
How IT is feeling.
important news/subject I am interested
remind me on things. I asked it to remember
tech issues
Stock market analysis
recommendations
music
What are some phrases your ideal conversational agent might say? 💬
Polite responses
Salutations
Friendly reminders
small bits of info such as weather
Just casual chat, like you do with your friends
Encouraging conversions
Start a really big argument with me.
What are some things your ideal conversational agent can do (e.g., tasks)? 🛠
comparision shop
Check your wellbeings
Buy groceries
Get help emergency's
Make recommendations based on your biometric data.
Suggest options for requests
Video game partner
Impersonate my voice, and get on the phone with people I know.
Send the car to get gassed up.
voice-controlled while I am driving, etc
play podcast
Deal with customer service bots for me
control smart home devices
Change voice or tone of voice for a more human like conversation
Generate new art, music, and movies based on my prompts.
What might your ideal conversational agent look like? 👀
Microsoft paper clip
Robots like Lovot
R2D2
Alien
Your favourite animal
talk to the same agent from every where (home, cars)
Butler for the family - shared info among family member, but with person
Gort -- the robot from The Day The Earth Stood Still
Frankenstein
A rock.
A chair with headphones
Other ideas 🚀

Appendix D

Additional Results from the Conversational Agent Development and Perceptions Study

This appendix contains additional information about responses to the surveys in Appendix B from the study discussed in Chapter 3. In particular, it includes information about participants' responses when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The figures in this appendix show the results from subsets of participants in this order: overall participants, participants from non-WEIRD countries, participants from WEIRD countries, children and then parents. For each figure, the technology (e.g., Google, the newspaper or Alexa) or people group (e.g., parents or friends), which I will call "systems" in this appendix, are ordered from greatest to least trust. Between each system there is a symbol indicating whether participants' trust in the system on the left is significantly greater than participants' trust in the system on the right. Table D.1 shows what each symbol denotes. For example, if Alexa was trusted significantly more than parents with a p-value of less than 0.001, the figure would show, "Alexa ***$>$ Parents"; alternatively, if Google was not trusted significantly more than the newspaper (but Google still had a greater mean level of trust), the figure would show, "Google ($>$) Newspaper".

Note that these figures only show significant differences between systems that are trusted most closely. For instance, in the pre-survey results in Figure D-1, Google and Alexa are compared, since participants trusted them to a similar extent, but Google and

Table D.1: Symbols from figures and their corresponding p-values.

Significance	Symbol
$p \leq .05$	*
$p \leq .01$	**
$p \leq .001$	***
Not significant	( )

parents are not compared (despite there being a significant difference between the two). In general, participants trusted technology (Google, the newspaper and Alexa) more than people (parents and friends).

Overall

Pre-survey:

Google, 3.96 (>) Alexa, 3.85 (=) newspaper, 3.85 ***> parents, 3.37 (>) friend, 3.04

Mid-survey:

Google, 4.00 (>) newspaper, 3.98 (>) Alexa, 3.88 **> parents, 3.38 ***> friend, 2.93

Post-survey

Google, 3.91 (>) newspaper, 3.94 (>) Alexa, 3.89 **> parents, 3.37 *> friend, 3.09

Figure D-1: The mean responses for participants overall when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The systems are ordered from greatest to least trust. Table D.1 shows the significance symbols.

Non-WEIRD

Pre-survey:

Google, 3.86 (>) Alexa, 3.76 (=) newspaper, 3.71 (>) parents, 3.38 *> friend, 2.95

Mid-survey:

Newspaper, 4.06 (>) Alexa, 3.94 (>) Google, 3.89 (>) parents, 3.50 **> friend, 2.94

Post-survey

Alexa, 3.94 (=) Google, 3.94 (=) newspaper, 3.94 *> parents, 3.47 (>) friend, 3.18

Figure D-2: The mean responses for participants from non-WEIRD countries when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The systems are ordered from greatest to least trust. Table D.1 shows the significance symbols.

WEIRD

Pre-survey:

Google, 4.04 (>) newspaper, 3.96 (>) Alexa, 3.92 **> parents, 3.36 (>) friend, 3.12

Mid-survey:

Google, 4.09 (>) newspaper, 3.91 (>) Alexa, 3.82 *> parents, 3.27 *> friend, 2.91

Post-survey

newspaper, 3.94 (>) Google, 3.89 (>) Alexa, 3.83 *> parents, 3.28 (>) friend, 3.00

Figure D-3: The mean responses for participants from WEIRD countries when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The systems are ordered from greatest to least trust. Table D.1 shows the significance symbols.

Children

Pre-survey:

Google, 4.04 (>) newspaper, 3.96 (>) Alexa, 3.93 **> parents, 3.41 *> friend, 3.00

Mid-survey:

Google, 4.17 (>) newspaper, 4.04 (=) Alexa, 4.04 **> parents, 3.46 *> friend, 2.96

Post-survey

Google, 4.05 (>) Alexa, 3.95 (>) newspaper, 3.90 (>) parents, 3.38 (>) friend, 3.10

Figure D-4: The mean responses for children when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The systems are ordered from greatest to least trust. Table D.1 shows the significance symbols.

Parents

Pre-survey:

Google, 3.84 (>) Alexa, 3.74 (>) newspaper, 3.68 (>) parents, 3.32 (>) friend, 3.11

Mid-survey:

Newspaper, 3.88 (>) Google, 3.75 (>) Alexa, 3.63 (>) parents, 3.25 *> friend, 2.88

Post-survey

Newspaper, 4.00 (>) Alexa, 3.79 (>) Google, 3.71 (>) parents, 3.36 (>) friend, 3.07

Figure D-5: The mean responses for parents when rating Google, the newspaper, Alexa, parents and friends on a 5-point scale in terms of trust of information correctness. The systems are ordered from greatest to least trust. Table D.1 shows the significance symbols.

Appendix E

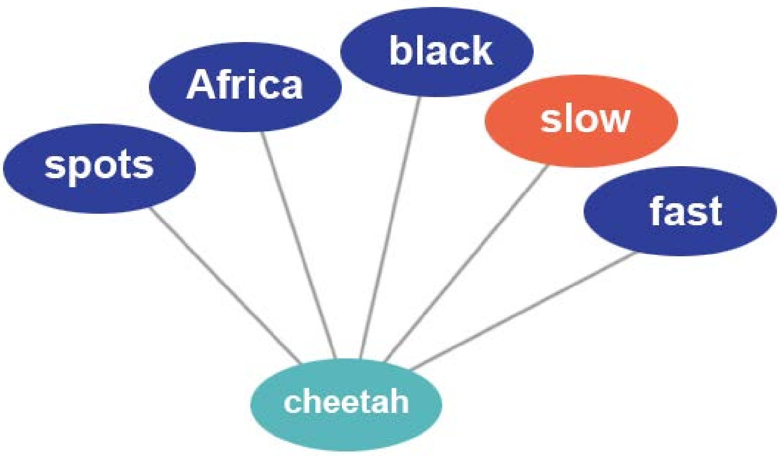
Zhorai Teaching Resources

This appendix contains teaching resources for the Zhorai activity [99], including a student worksheet, worksheet rubric and teacher guide. The Zhorai repository is stored on GitHub [192].

Name: ____________________

What have we learned from Zhorai?

1. Which sentences could you say to Zhorai to create the following mind map?



____

____

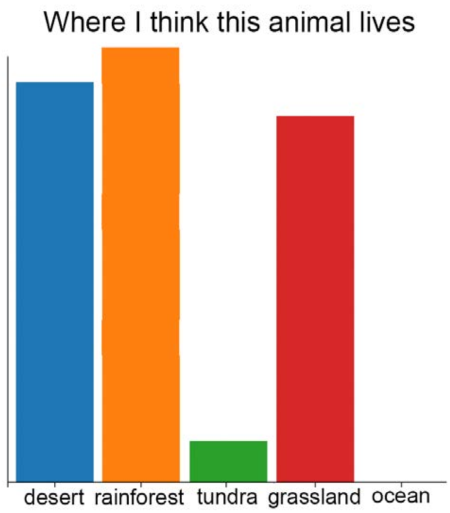
____

____

____

2. What could you tell Zhorai about monkeys so that it could correctly guess that monkeys live in rainforests? Try to use at least three sentences.

3. The following histogram is what Zhorai thinks about where “toucans” live. Based on this histogram, which ecosystem would Zhorai think a toucan lives in?



4. Which ecosystems do snakes live in? Why might Zhorai have a difficult time classifying snakes into one ecosystem, even if it knew everything there is to know about them?

---

5. Have you tried saying “Zhorai” to Zhorai? If not, ask the teacher if you can try. Does Zhorai recognize its own name? If not, why do you think it doesn’t? Can you think of another name that Zhorai won’t recognize?

---

6. After teaching Zhorai new information, what would you like to teach Siri, Alexa, or any other conversational agent about? How might you teach it?

Worksheet Rubric

Question 1		
Goal	**Evidence: I see students...**	**Notes**
Students understand mindmap is about cheetahs	• Using at least one sentence with "Cheetah(s) is/are..."	
Students understand the colors of the mindmap	• 4 - Gets all sentences correct. E.g., "Cheetahs live in Africa" "Cheetahs are fast" "Cheetahs have black spots" "Cheetahs are not slow" • 3 - Misses one of the sentences or uses "Cheetahs are black" "Cheetahs are slow" • 2 - Only gets two sentences correct • 1 - Only gets one sentence correct	
Question 2		
Goal	**Evidence: I see students...**	**Notes**
Students understand how sentences are created	• Using at least one sentence that describes monkeys (e.g. "Monkey(s) is/are..."	
Students understand that the sentences need to incorporate details about the rainforest	• Using at least one sentence that logically references the rainforest mindmap (e.g. has words like "trees", "plants", "rain", "humid", "damp", etc) • 4 - Uses three or more sentences that logically refer to the rainforest mindmap • 3 - Uses two or more sentences that logically refer to the rainforest mindmap • 2 - Uses at least one sentences that logically refer to the rainforest mindmap • 1 - Uses at least one sentence that describes monkeys (e.g. "Monkey(s) is/are	
Question 3		
Goal	**Evidence: I see students...**	**Notes**
Students understand what the histogram means	• Choosing "rainforest" as the ecosystem Zhorai thinks a toucan would live in	
Question 4		

Goal	Evidence: I see students...	Notes
Students recognize snakes live in many different ecosystems	• Listing various ecosystems that snakes live in • If they do not understand that snakes like in multiple places, then this is invalid	
Students relate to how Zhorai could still guess “incorrectly” because there are multiple correct answers in this case	• Providing at least one reason that is similar to “Zhorai would be confused because there is no definitive classification for snakes”	
Question 5		
Goal	**Evidence: I see students...**	**Notes**
Students see that Zhorai does not recognize its own name	• Answering “no” to “Does Zhorai recognize its own name?”	
Students understand that Zhorai only knows English words	• Providing at least one plausible reason • Providing another name/word that is not an English word	

Zhorai - Elementary School - Grades 3-5

Essential Question

How can an AI learn through interaction?

Objectives

Students will be able to

- Interact with a conversational agent
- Identify what AI can learn
- Brainstorm implications of bias in AI

Connection to Common Core

ISTE standards for CS Educators

- Standard 1. Computational Thinking - Learner (Understand the software component of computing systems)
- Standard 2. Computational Thinker - Facilitator (Educators engage students in identifying problems that can be solved using computational thinking)
- Standard 6. Equity and Inclusion - Advocate (Address a diverse range of ethical, cultural, and social perspectives on computing)

CSTA K-12 CS Standards (Grade 3-5)

- 1B-IC-18 - Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices.
- 1B-CS-01 - Describe how internal and external parts of computing devices function to form a system.

Digital Literacy and CS (Massachusetts only)

- how computing affects society (for example, privacy and the security of information)

Tools and Materials

- Laptop (3-5 students per device per facilitator)
- Pencils (1 per student)
- Pre-activity assessment
- Post-activity assessment
- Worksheet
- Teacher resources

Summary

This small-group, learning activity focuses on learning how an AI can learn through teaching a conversational agent. As a demonstration, this lesson allows the teacher and students to explore and experiment with teachable AIs using a voice-based input and a browser-based UI that displays the mental model of the AI. The teacher will explain what to teach Zhorai, show what Zhorai has learned, and facilitate student discussion on how Zhorai can be biased. The lesson will conclude by asking students to imagine what happens if AIs are biased in the real world, and how that would impact society.

Agenda

- Warm-up (10 minutes)
- Module 1: Understanding Knowledge Representation in Machines (10 minutes)
- Module 2: Teaching New Knowledge to Machines (10 minutes)
- Module 3: Witnessing Machine Learning Classification (10 minutes)
- Module 4: AI & Ethics Discussion (15 minutes)
- Cool down (5 minutes)

Assessment

Demonstration of Learning

- Students can interact naturally with Zhorai
- Students can understand how Zhorai organizes its knowledge (i.e., knowledge representation)
- Students teach Zhorai new knowledge

Classroom Discussion

- Student-generated ideas of what to teach Zhorai (i.e., what facts to feed to Zhorai's training model)

Oral Responses

- Students can explain how Zhorai makes decisions in Module 2
- Students can explain how AI (like Zhorai) can have bias and real-life implications of bias

Written Responses

- Students complete pre-activity worksheet to demonstrate any existing knowledge about AI and ML
- Students complete post-activity worksheet to demonstrate mastery of learning activity concepts

Teaching Guide

Warm-up (10 minutes)

Teacher begins class by assessing students what they know about learning in AI.
Potential discussion questions:

- How do you think Alexa or Siri works?
- Can Alexa learn new things?
- Can we teach Alexa new things?

Teacher then uses computer to introduce Zhorai. Zhorai has a built-in self-introduction module where Zhorai introduces what planet they're from (i.e., small talk) and students have a chance to build rapport with Zhorai the fictional character. The following script is an example of what this dialog might look like:

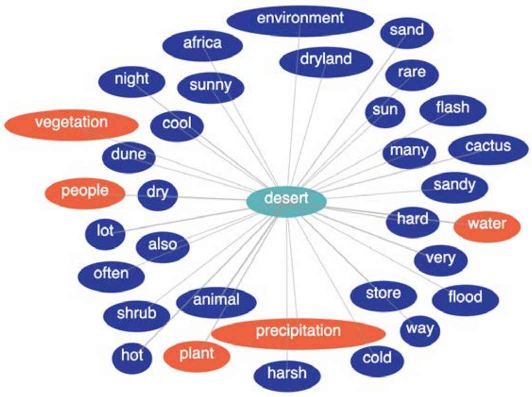
Moderator: Today we're going to talk to an alien from another planet! It's name is Zhorai, and it's actually a computer program, or *conversational agent* that MIT students created. Would someone like to say hi to Zhorai?
(The moderator chooses a student, who comes up to the computer and clicks record.)
Student: Hi!
Zhorai: Hi there! What's your name?
Student: My name is Erin.
Zhorai: Erin, what a nice name. Where are you from?
Student: I'm from Boston.
Zhorai: Interesting! I'm from planet Igbruhmmelkin. I've never heard of Boston before. Can you tell me more?
(A button appears with the words, "Find out what Zhorai knows about your planet!", which takes the class to the next module.)

Module 1

1. After the students have a chance to interact with Zhorai and familiarize with the controls (record and listen buttons), the teacher then states, "This is Zhorai's first time on Earth and Zhorai made a lot of observations about Earth's ecosystems. Does everyone know what an ecosystem is? (Zhorai knows about the desert, ocean, grassland, forest, and tundra). Let's look inside Zhorai's brain and see how it thinks!
2. The teacher guides students into making observations about the ecosystem mindmaps. The teacher records student asking, "Zhorai, what do you know about deserts?"
3. After each ecosystem, the teacher shows the students what Zhorai's mental representation of each ecosystem is (i.e., the mindmaps).
4. The teacher shows them the sentences that Zhorai's model was trained on and asks, "What do you notice about these sentences and how it connects to Zhorai's mindmap for [ecosystem]?"

The following dialog illustrates what Module 1 might look like:

Moderator: Zhorai already knows a little bit about planet Earth. Let's ask it about what it knows. Who wants to ask Zhorai what it knows about the desert?
Student: What do you know about deserts?
Zhorai: Here's what I know about deserts. They're fascinating! (A mindmap about deserts appears, as shown below. This mindmap was created using the sentences that appear below the mindmap.)



The desert mindmap.

Zhorai heard these things about deserts from other Earthlings:

The desert has lots of sand
The desert is very dry
Deserts have cactus
The desert is very hot and sunny
Deserts don't have much water
Deserts have few people
Deserts have flash floods

A few of the sentences that created the desert mindmap.

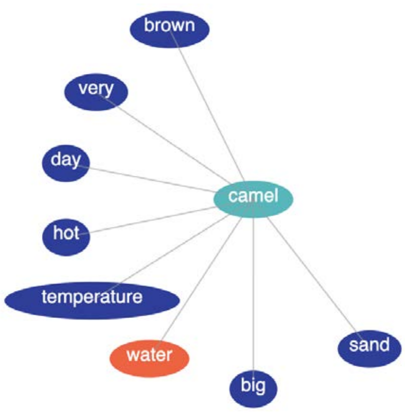
Moderator: Why don't we ask Zhorai what it knows about oceans?
Student: What is an ocean?
Zhorai: Oh yes, oceans are very interesting. Here's what I know. (Another mindmap and list of sentences appears.)
Moderator: What other ecosystem can we ask Zhorai about?
Student: What do you know about plains?
Zhorai: Hmmm, I haven't heard about that ecosystem before, but I know about rainforests.
The class discusses why Zhorai knows only about some ecosystems. This is because Zhorai was only given data about specific ecosystems. In other words, Zhorai's *machine learning model* was only *trained on* particular data.

Module 2

1. The teacher introduces the second module by asking students, "Let's think about an animal that lives in one of these ecosystems. How might Zhorai know where this animal lives without us telling Zhorai explicitly?"

2. Let's try it! Teacher guides students into describing animals without saying the animal name (i.e., Taboo).
    a. Here are some example sentences that enable Zhorai to classify the animal correctly. Notice how descriptive they are. Try to have students talk about the climate and vegetation that the animal lives around.
        i. Bees fly around from plant to plant spreading pollen
        ii. Birds fly from tree to tree
        iii. Camels live in hot dry places with lots of sand and they don't need much water
        iv. Cows graze in fields and eat lots of grass
        v. Dolphins swim around in the sea and blow water out of their blowholes
        vi. Fish are found in wet places with seaweed
        vii. Polar bears live near icebergs in the arctic
        viii. Whales are the largest animal in the sea and they blow water out of their blowhole
3. Afterwards, teacher shows the students the mindmap that Zhorai builds for the animal they discuss.

Moderator: Zhorai knows a little bit about ecosystems, but doesn't know about what animals live in each ecosystem. Let's teach Zhorai about these animals.
(In this module, there is a prompt on-screen that says, "Zhorai would like to know about *camels*. Could you teach it about them?".)
Student: Camels are big and brown. They walk on the sand all day. Camels live in hot temperatures and they drink very little water.
Zhorai: Wow, camels sound really interesting! Let me think for a bit and then I'll show you my thoughts.
(A mindmap about camels appears on screen, as shown below.)



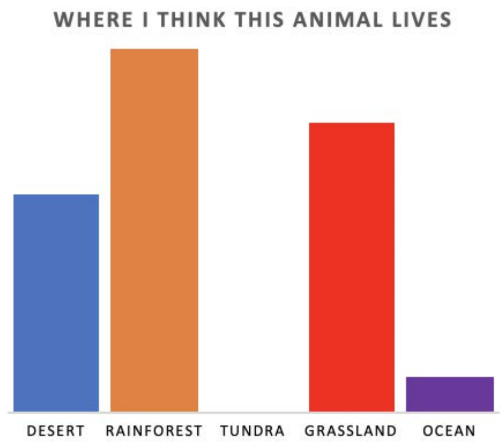
An example mindmap for camels.

(A new prompt appears on the screen stating, "Zhorai would like to know about *bees*. Could you teach it about them?".)
Student: Bees make honey. They are very small and collect pollen from flowers every day.
Zhorai: Bees sound fascinating! Now I want to visit earth and all of it's life! I'll show you what I understand after I think for a little while.
The students then discuss how well they think Zhorai understands each animal.

Module 3

1. Teacher introduces third module by asking students, “Now, let’s see how Zhorai makes decisions on which ecosystem [animal] lives in!”
2. Teacher shows the students the bar graph Zhorai builds for comparing how close an animal is to the ecosystem representation. The students should notice that the Zhorai chooses the ecosystem that is the closest in proximity to the animal’s datapoint.

The following script illustrates example dialog for Module 3:
Moderator: Now that Zhorai knows about ecosystems and animals, let's see if it can guess which ecosystems certain animals belong to. Who would like to ask Zhorai to guess?
Student: Where are bees from?
Zhorai: Oh yeah, bees sound interesting. Let me think about where they might be from.
Zhorai: Based on what I know about Earth, I would guess bees live in rainforests. (A bar graph appears on screen with the rainforest bar as the highest, as shown below. The mindmap about bees from Module 2 also appears on screen.)



An example bar graph for bees.

Moderator: Let's think about this for a second. Zhorai is a computer program, so it uses numbers to think. This scatter plot shows what Zhorai understands about ecosystems and fish. You can see that there are numbers on the sides of the graph. Based on this graph, why do you think Zhorai thinks bees live in rainforests?
Student: Because the bar for rainforests is the highest on the graph!
Moderator: Exactly! The similarity (which can be represented by a number) between bees and the rainforest ecosystem is the highest. Let's see if Zhorai can guess where another animal comes from.
Student: Guess where polar bears are from.
Zhorai: I'll think about polar bears for a bit and let you know!
Zhorai: Hmm, I'm not sure! I haven't heard much about that animal.
Moderator: Since we didn't teach Zhorai about polar bears, it doesn't have any idea where they are from.

Each classification will be accompanied by a discussion about how Zhorai represents knowledge and why it has succeeded or failed at the classification task. Zhorai may fail at the classification task if it is not given enough descriptors to place the animal correctly in the *embedding space* (which is the high dimensional space that is simplified and illustrated by the scatter-plot). Zhorai needs a sufficient amount of description words about animals to discern where the animals might live.

Module 4

1. Teacher can discuss how Zhorai works:
    a. **How does Zhorai learn and perceive the world?**
        i. Zhorai learns through being given information. It does not know anything without being given information. For example, Zhorai only knows about ecosystems because it was previously given information from the internet about ecosystems. Since Zhorai was not given information specifically about animals, it only knows about animals if it is told about them.
        ii. Zhorai can perceive the world by using a microphone. The microphone converts the sound waves to numbers, the numbers to phonemes, the phonemes to strings of letters and words, and the words to sentences. The sentences are then *parsed* or sectioned into parts and can be represented by a mindmap.
    b. **How did Zhorai learn English?**
        i. Zhorai learned English in a similar way to how it learned about ecosystems: It was given many examples of English words being spoken. Through the audio files and transcriptions of the words being spoken, it learned how to replicate the audio into realistic-sounding speech. The transcriptions are usually manually created by hand, so the computer is very dependent on humans to learn how to speak.

c. **Is it a similar way to how it learned about ecosystems?**
    i. Yes, it is similar because the system required a lot of data (speech examples, in this case) to learn how to speak. The system also required a lot of ecosystem data to learn about Earth.
d. **How did Zhorai hear us?**
    i. Zhorai heard us through a microphone, which converts sound waves to digital signals and numbers.
e. **What does the computer "see" or perceive when it gets information from a microphone?**
    i. The computer "sees" an array of numbers. You can imagine this as a graph with time on the x-axis and pitch (or sound) on the y-axis. The resulting graph looks like a "wave". This is what we call a "sound-wave".

2. Teacher can discuss how Zhorai learns:
    a. **What else could we teach Zhorai?**
        i. Zhorai is currently programmed to learn only about ecosystems and animals, but it could be programmed to learn about anything, like common conversational agents, such as Siri and Alexa. We could teach these conversational agents about anything.
    b. **Can we teach it about different cultures?**
        i. Sure, why not?
    c. **How would that be different?**
        i. We would be talking about people and their cultures as opposed to animals and their environments. Then, biases would be much more apparent.
    d. **Would Zhorai know whether what we teach it is correct?**
        i. No, Zhorai would not know whether we teach it is correct unless we told it (or programmed it to know) that it is not correct. For example, we could teach it that polar bears are tiny animals that like to eat rocks or we could teach it that Canadians are people who drink maple syrup for breakfast.
    e. **How would you feel if Zhorai learned something untrue about your culture?**
        i. For example, how might you feel if Zhorai learned that all Americans are mean and nasty people?
        ii. Some artificial intelligence agents have *biases*, just like humans do, since they learned from human-created data. It is important for the developers of these systems to minimize these biases to ensure all people are treated equally by the systems.
3. Teacher can discuss implications of Zhorai's mistakes:
    a. **Why were there animals whose ecosystem Zhorai could not guess?**
        i. Maybe some animals weren't explained thoroughly
        ii. Maybe some animals live in more than one ecosystem
        iii. Maybe Zhorai does not know enough about a certain ecosystem
    b. **Is Zhorai's knowledge of the world biased?**

i. Yes. First of all, Zhorai only knows about the ecosystems that were taught to it. It cannot classify animals to ecosystems that were not taught to it.
ii. In addition, there is bias between the ecosystems that it knows. It is possible that Zhorai knows a lot more about one ecosystem than the other. In that case, if an animal isn't explained thoroughly, it is more probable that Zhorai will guess one ecosystem over another. If the amount of knowledge about each ecosystem isn't the same, then they don't all have a fair shot at being the prediction when the prediction is unclear.
iii. Lastly, Zhorai's knowledge of animals is biased to what has been said to it. For example, if students only talk about the what the animal eats, then Zhorai will be able to classify better ecosystems with clear vegetation and animals with a clear diet.

c. **How can we improve this?**
    i. By training Zhorai with much more varied data, we can minimize Zhorai's biases.
d. **Why is it important?**
    i. It might not seem so important that Zhorai is biased towards ecosystems or animals, but if we generalize this to AI used in our day to day life, we can understand how biased systems can affect society.
e. **How can we cause Zhorai to make mistakes?**
    i. If we teach Zhorai incorrect details about animals, it will classify the animals incorrectly. For example if we say that fish live in trees, Zhorai might guess that fish live in rainforests.

4. Teachers discuss with students the ethical implications of what Zhorai knows and how Zhorai learns.
    a. In continuation to the previous questions, explain how biased systems affect society. Explain why it is important that they are trained with data that represents all of society and not only parts so that they can work just as well for everyone.

Cool down (5 minutes)

Teacher ends class by assessing students what they have learned in the last hour.

- Use post-learning activity assessment worksheet.
- Other possible discussion questions:
    - How is Alexa different from Zhorai?
        - Alexa was trained and continues to be trained on corpuses of all different kinds of areas so that Alexa know not only about ecosystems, but about many concepts that humans discuss.
    - How many hours do you think people needed to talk to Alexa to get as good as it is now?
        - This is an open ended question, but Alexa has been trained on corpuses with billions of sentences.

    - How many people do you think talked to Alexa? Who do you think talked to Alexa?
        - In 2018, 39 million Americans owned Alexa. Alexa is regularly used by more than one user in a home. Also, these are only the American users.
        - All sorts of people talk to Alexa about all sorts of things. This is how Alexa knows so much about so many subjects.

External Resources

Synthesizing speech

Machine learning and how computers learn to understand speech

How does Alexa work?

What is a semantic parser?

What is a word embedding?

Appendix F

ConvoBlocks Teaching Resources

This appendix contains teaching resources for the ConvoBlocks activity [185, 186, 193, 191], including an example tutorial, agenda for a two-day workshop and teacher guide for a five-day workshop. The ConvoBlocks repository is stored on GitHub [194].

Carbon Info: A Simple Alexa Skill

In this tutorial, you will learn how to use the **MIT App Inventor's Conversational AI Interface** to create your own Amazon Alexa skill. You will also learn how to test your new Alexa skill using Amazon's **Alexa Development Console**.

> **What is this box?**
>
> Throughout this tutorial, you might see boxes like this one. These boxes will give definitions and explanations for keywords that are used when creating an Alexa Skill.

Create an Amazon Developer Account

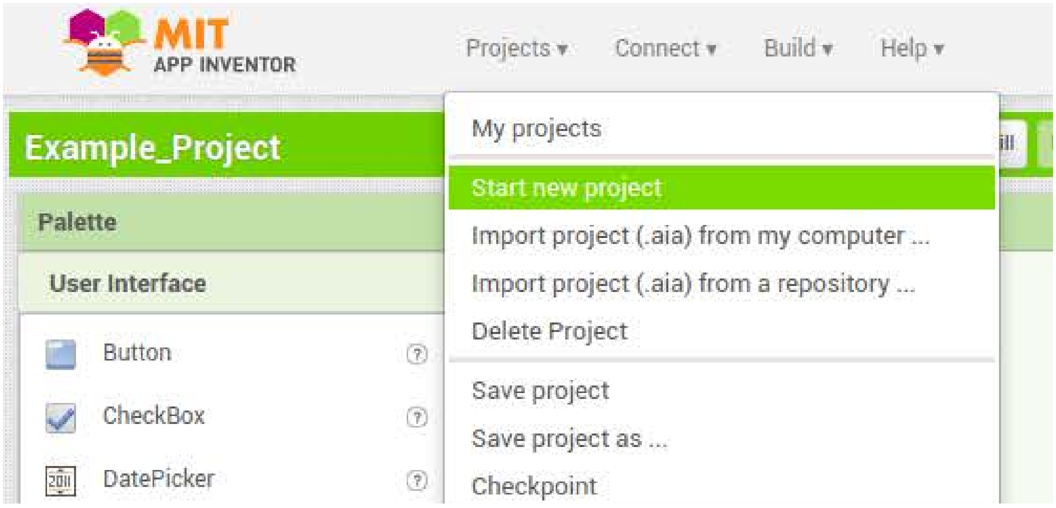
If you haven't already, follow the instructions in the *Create an Amazon Developer Account* document to create a free **Amazon Developer Account**. This will allow you to save any Alexa Skills to your account and use your skills on any Alexa-enabled devices that you have linked to your Amazon account.

Create a New Project

Navigate to: https://alexa.appinventor.mit.edu.

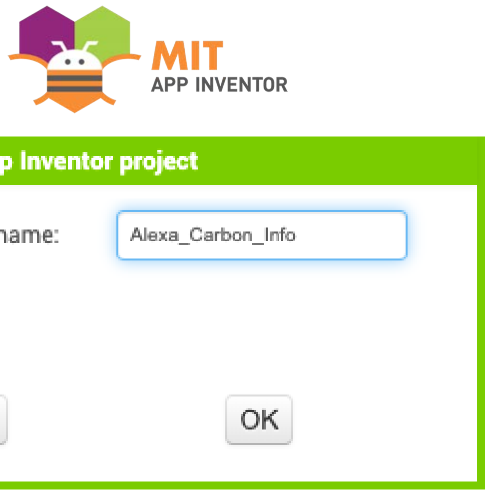
> **The Alexa Editor Website!**
>
> Notice that https://alexa.appinventor.mit.edu is not actually the regular App Inventor website! Make sure you use this link, else you won't find the option to add an Alexa Skills.

If you have another project open, go to the **Projects** menu and choose **New Project**.



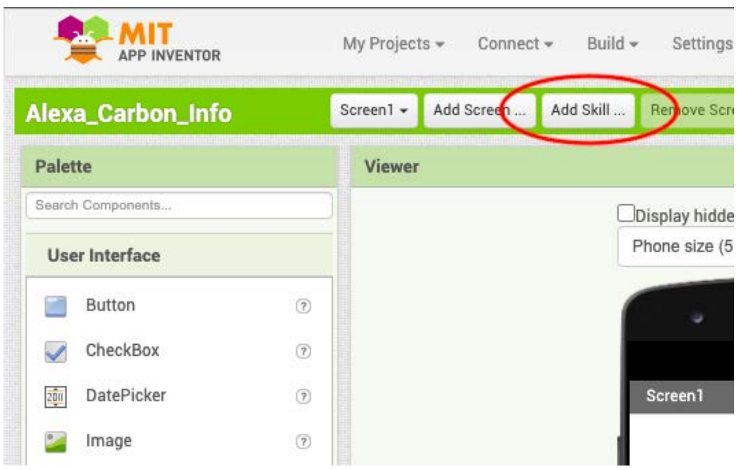
Name the Project

You can give your projects any name you like as long as there are no spaces (underscores are completely fine though). For this demonstration, let's name the project *"Alexa_Carbon_Info"*.

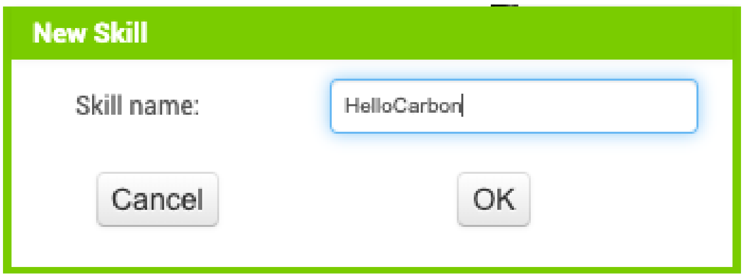


Add a New Skill

In the Designer Toolbar at the top of the screen, you will see a button labeled **“Add Skill …”**. If you don’t see this button, you may be on the wrong MIT App Inventor page. Make sure you’re using alexa.appinventor.mit.edu and *not* appinventor.mit.edu.

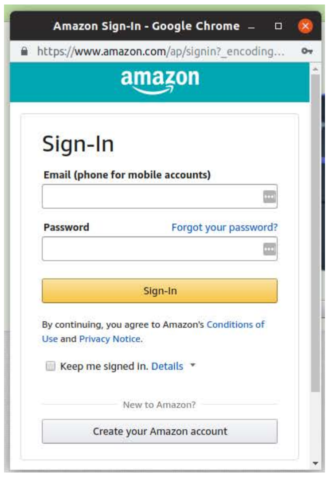


Click on the **Add Skill** button and enter the name of your skill. For this demonstration, we will name this skill *"HelloCarbon"*.



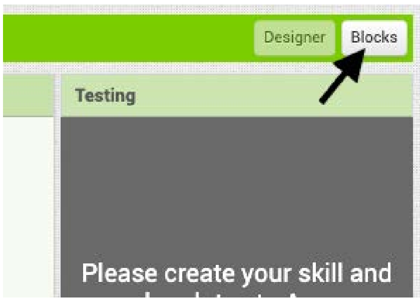
Login to your Amazon Account

You will now be taken to a new Designer Page for the Conversational AI Interface. Under the image of the Amazon Echo Dot, there should be a button labeled **Login to Amazon**. Click that button and enter your Amazon Developer Account information into the external pop-up. *If this window does not appear, check if your browser has blocked a pop up and allow the pop-up.*



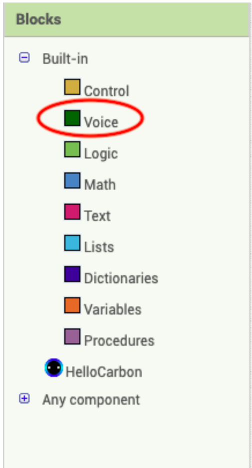
Open the Blocks Editor

After you've successfully signed in, switch to the **Blocks Editor** by clicking the **Blocks** button in the top right.



Notice the New Components

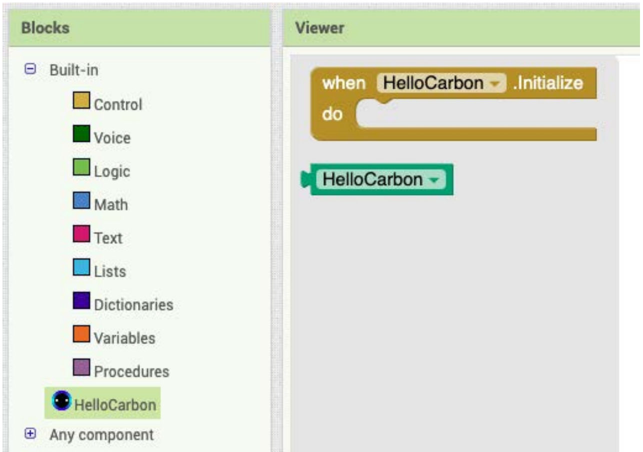
Notice that in the menu on the right, there is a new built-in drawer labeled **Voice**.



There's also a component below it with the same name as your skill. This is your **Skill Component**.

Add the Initialize Block

Click on the **Skill Component** at the bottom of the menu. This will open a *blocks* drawer.



Drag the **Initialize Block** onto the workspace.

Everything inside this block will tell Alexa the structure of your Alexa Skill and all of the phrases it should be listening for.

Add the Invocation Name

The first thing we want to teach our skill to listen out for is its **Invocation Name**.

What is an Invocation Name?

Just like how every mobile app needs to have a name, so does our custom Alexa Skill. An **Invocation Name** is just the name of the Skill that we are making, and is used to “invoke” our skill. The structure of any command you will tell Alexa is:

“Alexa, tell <Invocation Name> to …”

The invocation name is what will help Alexa tell which Skill it needs to use, so make sure that every skill you make has a *unique* name.

For example, if we decided to make the Invocation name of a custom skill “Codi bee” we would say:

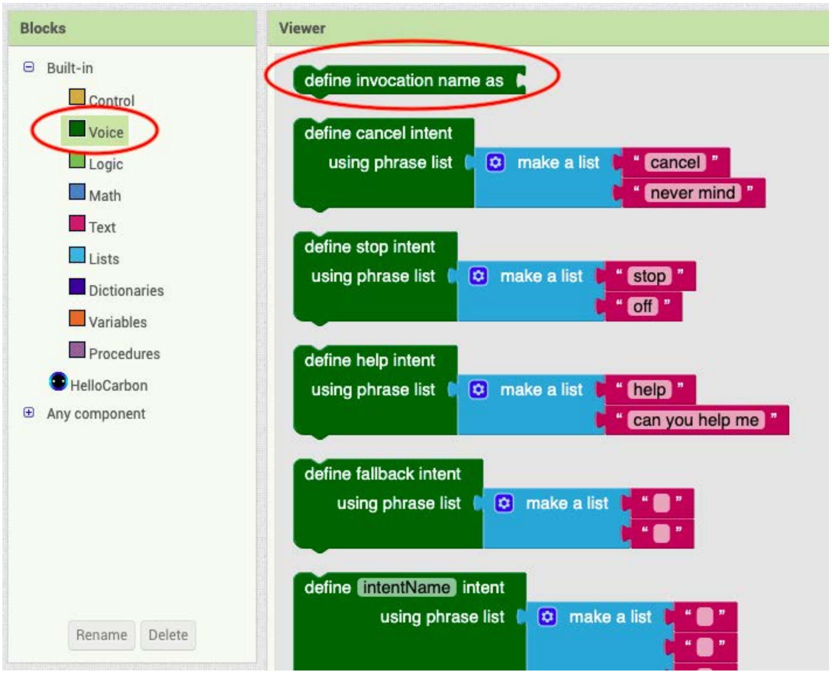
“Alexa, tell Codi bee to do something”

Note: The invocation name needs to be at least two (2) words long, but avoid making it a full sentence, since you will be saying the name a lot.

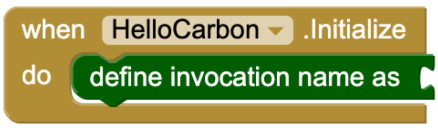
For this demonstration, let’s name this skill “Hello Carbon”. This means that when we want to call our skill, we will say:

“Alexa, tell Hello Carbon to do something”

To define this invocation name, open the **Voice** blocks drawer. The topmost block is the **define invocation name** block.



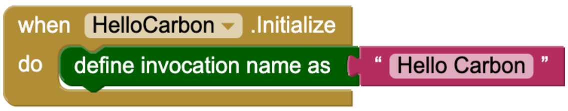
Drag the **define invocation name** block inside the **when initialize** block.



Now get a Text block from the Text drawer and add it to the invocation name.

Carbon Info: A Simple Alexa Skill App - 9

Don't mind the error for now. It's just a warning saying that the invocation name should not be empty. Inside the text block, enter your invocation name. *Remember, two or more words, but not a full sentence!* For this tutorial, we will make this invocation name "Hello Carbon."



Define a Custom Intent

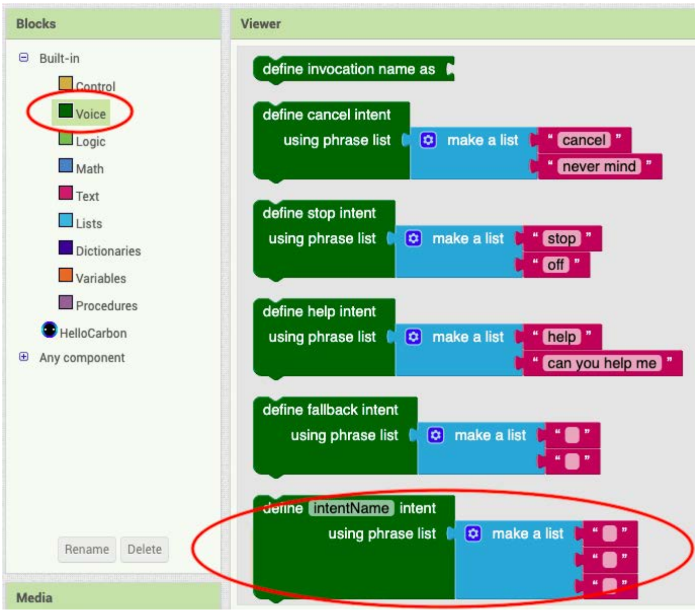
Now we need to add a custom **Intent** to our app.

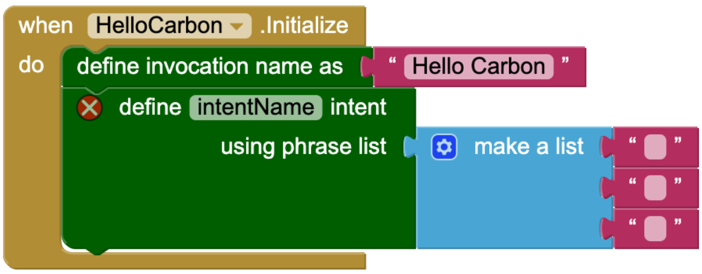
What is an Intent?

You can think of an **Intent** as a "command" that you want to teach Alexa. It could be a Stop Intent, a SayHelloWorld Intent, or any intent you can really think of that you will need in any Skill.

Carbon Info: A Simple Alexa Skill App - 10

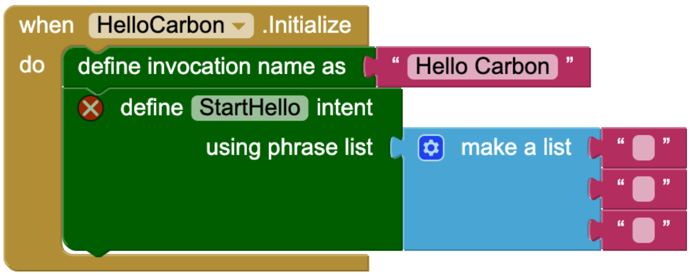
Let’s teach our app to listen for the “TalkAboutCarbonFootprint” intent. To define this intent, open the **Voice** drawer again and scroll down to the block that says **Define intentName Intent**. Drag this **Define Intent** block under the invocation name.





Carbon Info: A Simple Alexa Skill App - 11

For this intent, we'll rename the intent to be *"StartHello"*. You can give this intent any name that makes sense to you, as long as it is a meaningful name.



Add a List of Utterances

As you might have noticed, we need to give the intent block a list of **utterances**.

> **What are Utterances?**
>
> Let's say your parent wants you to do the dishes. They might say "Clean the dishes." Or, maybe they'll say "*Do* the dishes." Maybe they'll just say "Wash the dirty plates." All these sentences have the same *intent*: you need to do the dishes.
>
> Just like this example, when talking to Alexa, there might be a lot of ways to say something to trigger an intent. Each of these phrases that all mean the same thing is called an **utterance**.

In these text blocks, we will write a few sentences which Alexa will understand to mean "StartHello". One sentence could just be "Talk about carbon footprint" and another sentence could be "Give me information on carbon footprint", or "Say something about carbon emissions".

(If you are familiar with the List block, you can make this list bigger by clicking the blue icon at the top left of the block and add as many phrases as you want. The more possible phrases you add, the more likely it is that Alexa will be able to understand you.)

Remember that the structure of any command you tell Alexa when you want to trigger a skill is:

"Alexa, tell <Invocation Name> to <Utterance>"

We already set the invocation name as Hello Carbon, and now we've added the intent phrases, "Talk about carbon footprint" and "Give me information on carbon footprint." Here is a full example of what Alexa might hear when we want to trigger our new Skill:

"Alexa, tell Hello Carbon to give me information on carbon footprint"

Update the Amazon Code

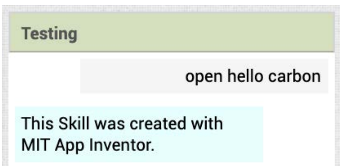
Now, we have finished telling Alexa the *structure* of our new Skill, also called the **Interaction Model**. This interaction model is made of the **Invocation Name** (Hello Carbon), the name of all the **intents** your skill needs to know, and all the **utterances** that will trigger each intent. Your blocks so far should look something like this.

Now, we need to send this **Interaction Model** to Amazon. Let's go back to the Designer page in the top right.

Below the picture of the Amazon Echo Dot, there is a button labeled **Send Updates**. (If you only see a button that says **Login to Amazon**, you need to do that first). Click this button.

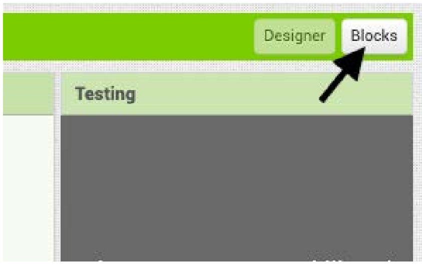
After a while, you should get a pop-up at the top of your browser that says "Skill updated successfully on Amazon".

After getting this message, you'll have to wait a bit longer for your skill to build, and then you'll be able to type things into the Testing Box and click Send. For example, you can type "Open Hello Carbon" and it should give you a response about how the skill was developed in App Inventor.

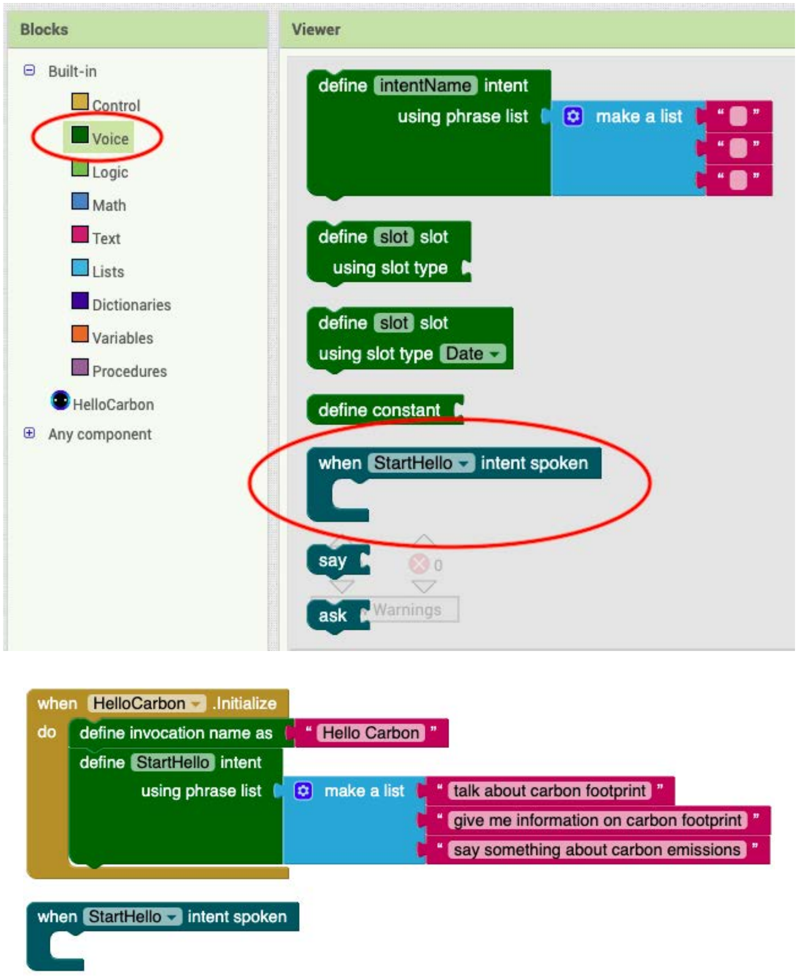


Define the Endpoint Function

Now that we have defined the Interaction Model and updated Alexa, we need to define the “brain” of our Alexa skill. Right now, our skill can recognize whenever we want it to “StartHello,” but it doesn’t know what to do or how to handle it. To make the response, we need to define the **Endpoint Function**, the “brain” of our Skill. To begin, let’s navigate back to the Blocks view.

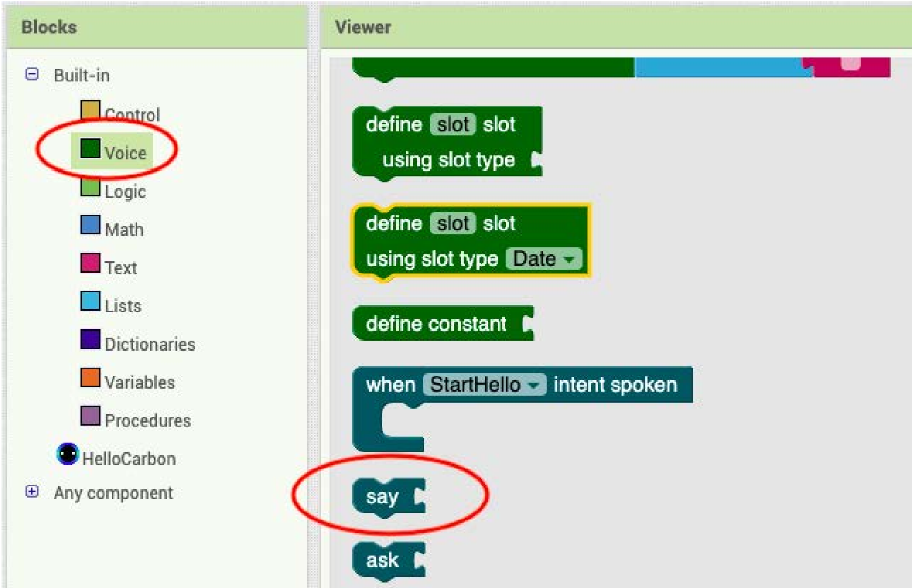


Open the **Voice** drawer and scroll down until you see the **when StartHello intent spoken** block. Drag this block below the When Skill Initialize block (not inside).



This block will determine what Alexa will do and say. You can add if statements and a lot of things you could do in normal App Inventor Procedures. For now, we simply want Alexa to say Hello World. To do this, we need to find the **Say** block in the **Voice** drawer.

MIT APP INVENTOR



Add the **Say** block to the inside of the **when intent spoken** block.

Next, attach a text block to the **Say** block and type inside the text block,“A carbon footprint is the total amount of greenhouse gases that are generated by our actions.”

Carbon Info: A Simple Alexa Skill App - 18

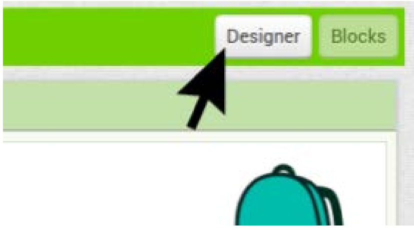
And now we've finished defining the Endpoint function! If you decide to add more intents, like a GiveMeFigures intent, you need to add a new **when intent spoken** block for each intent.

Tip on the Say block

When playing around with your Endpoint function, you may be tempted to add more Say blocks to an intent. But be careful! Alexa will only say whatever is in the *last* Say block of any intent.

Generate the Endpoint Function

Now, go back to the Designer view.



We want to send our new **Endpoint Function** to Amazon. Click the button labeled **Send Updates**.

Then wait for the skill to update and finish building.

Carbon Info: A Simple Alexa Skill App - 20

After a while, you should get a pop-up at the top of your browser that says "Skill updated successfully on Amazon".

After getting this message, you'll have to wait a bit longer for your skill to build, and then you'll be able to type things into the Testing Box. For example, you can type "Open Hello Carbon" and it should give you a response about how the skill was developed in App Inventor.

We have successfully made our first custom Alexa Skill!

If you decide to make any changes to your Alexa Skill after completing this tutorial (e.g., updating/adding/deleting blocks), make sure to do this step again to send your new changes to Amazon.

Test your Skill

Now it's time to test our Alexa skill! If you have an Alexa-enabled device handy, say "Alexa, tell Hello Carbon to talk about carbon footprint." to it and hear the response. You can also do this on your mobile device by downloading the free **Amazon Alexa** app, which comes with a built-in Alexa simulator.

If you don't have an Alexa-enabled device on hand, the Designer page in App Inventor allows you to simulate an Alexa using your custom Alexa skill right in your browser! Simply type in the textbox at the bottom, “Tell Hello Carbon to talk about carbon footprint.” (You can omit the Alexa at the beginning of the phrase when using the testing box). The response should be what we plugged into the **say** block earlier, “A carbon footprint is the total amount of greenhouse gases that are generated by our actions.”

Finish!

Congratulations! You’ve made your first custom Alexa Skill! Feel free to extend this app by adding new intents and new ways for Alexa to respond to each intent.

ConvoBlocks Day 1 & 2 Agenda

Agenda Day 1

Welcome to the Future Worlds Challenge! Day 1 will include learning 2 or 3 tutorials about how to program a **Carbon Footprint conversational agent** that runs on Amazon Alexa. Programs for Alexa are called "**Alexa Skills**".

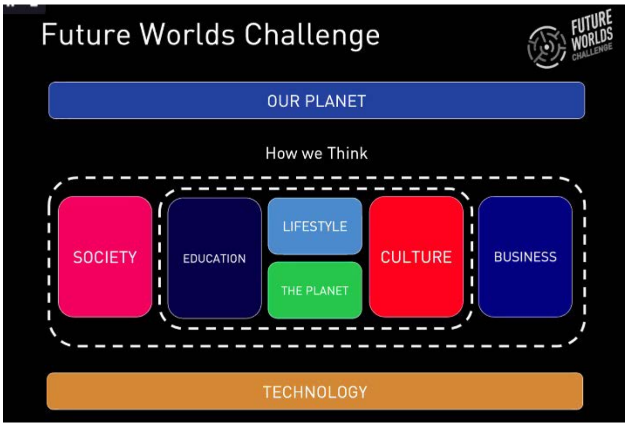
Agenda

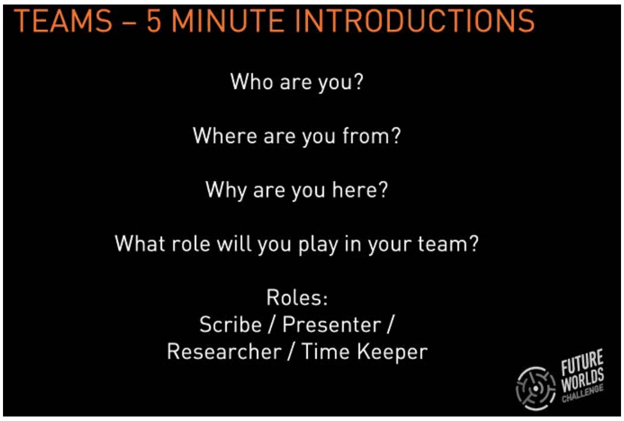
1. Open Zoom Room (4:30 am ET)
2. 👋🏾 Welcome! (5:00-5:10 am ET)
3. 📝 Pre-questionnaire (5:10-5:25 am ET)
    a. This is to get your initial thoughts on conversational agents like Alexa, so please don't look anything up online or talk to anyone about your answers— We want to hear from *you* 😊
4. 🌏 Tutorial 1: Carbon Info (5:25-6:00 am ET)
    a. "**Ask** Hello Carbon to talk about carbon footprints"
5. ☕ Quick break (6:00-6:05 am ET)
6. 💭 Idea time! How do you envision the future of conversational agents? (6:40-7:00 am)
    a. Miro Board
7. 👣 If time allows:
    a. Tutorial 2: Carbon Calculator (6:05-6:40 am ET)
8. ⏰ If time allows:
    a. Tutorial 3: Personal Footprint (7:00-7:30 am ET)
    b. Tutorial demo
9. 📝 Mid-questionnaire (7:30-7:45 am ET)
10. 🙋🏽‍♀️ Team formation (7:45-7:55 am ET):
    a. Scribe, Programmer, Presentation Overseer, Vision
    b. Parents: Logistics, journaling, and keeping everyone on track
11. 👋🏼 Close— See you tomorrow! (7:55-8:00 am ET)

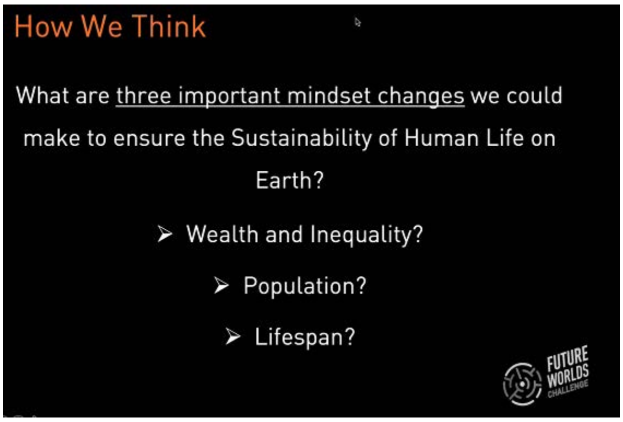
Agenda Day 2

Day 2 will involve coming up with ideas for the “Future Worlds Challenge”, creating an Alexa skill and doing a presentation with your team.

Agenda

1. Open Zoom Room (4:30 am ET)
2. 👋 Welcome back! Having Brave Conversations (5:00-5:05 am ET)
3. Session One (5:05-5:30 am ET)
    a. 

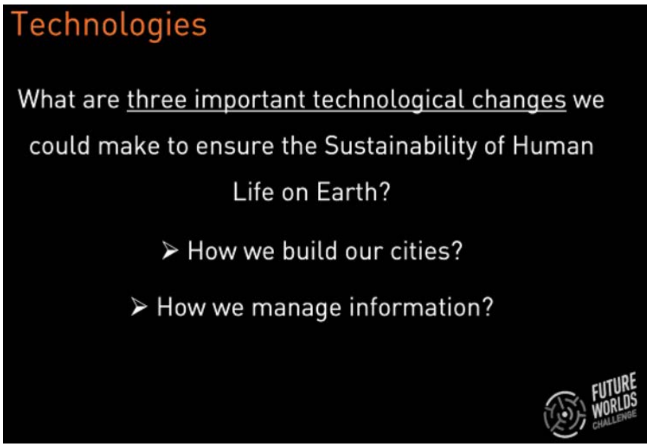
    b. 

c. 

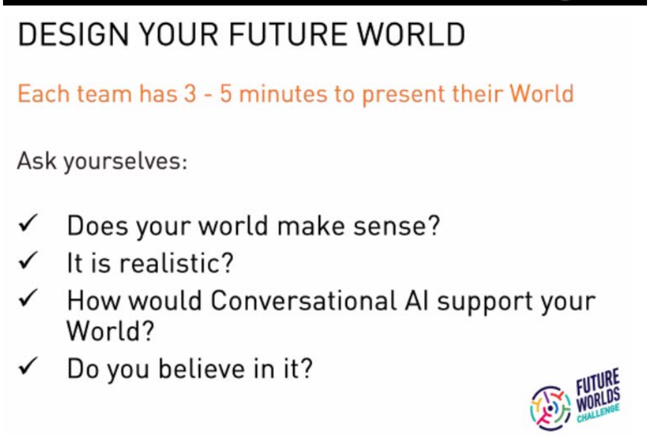
4. 🌍 Overview and Future Worlds Challenge (5:30-5:55 am ET)
5. ☕ Quick break (5:55-6:00 am ET)
6. Session Two (6:00-6:55 am ET)

a. 

7. ☕ Quick break (6:55-7:00 am ET)
8. 🎙️ Future Worlds Challenge Team Presentations (7:00-7:30 am ET)
    a. Presentations: 3-5 mins each
    b. Timeframe for your world: 10 years



c.



d.

9. 📝 Post-questionnaire while Judging and Reporting back (7:30-7:45 am ET)
10. 👣 Next Steps (7:45-7:55 am ET)
11. 👋🏿 Close— See you next time! (7:55-8:00 am ET)

Amazon Future Engineer Workshop: Teachers' Guide

Table of Contents

This document provides advice and a general guideline for teachers wishing to replicate the remote Amazon Future Engineer Conversational AI Workshop 2020, where we taught high school students how to create their own Alexa Skills through the App Inventor Alexa Interface.

Before Starting The Workshop

What is this box?

Throughout this guide, you'll see boxes like this one. These boxes will give definitions and explanations for topics discussed throughout this document.

Assumptions About Students

1. The students are not familiar with the MIT App Inventor interface. If all of the students are familiar with App Inventor, then Tutorial 1 can be skipped.
2. The students are in high-school / secondary school (or 8th to 12th grade) or can otherwise practice abstract thinking.
3. The students have completed the technical requirements prior to the workshops. Some requirements are specific to the nature of remote instruction.
4. The students have a gmail account they are comfortable with using for App Inventor
5. Students do not require an Amazon Alexa-enabled device to complete this workshop, but it is encouraged to have one. Alternatives include the Amazon Alexa mobile app.

Notes for the Teachers

1. In order to effectively guide students in developing their final projects, teachers need a broad understanding of the App Inventor programming interface.
2. Each student will need their own Amazon Developer account.
3. The curriculum is designed for five 2.5-hour workshop days. We recommend having a break sometime halfway through the workshop day

Day 1 - Introduction to App Inventor

The first day of the workshop has two main goals. First and foremost, we want to build familiarity with the App Inventor programming interface. Second, we want to plant the seed for discussions on Artificial Intelligence (AI) on Day 2 by building a very simple conversational agent (or computer program that can converse with a human).

To build familiarity with App Inventor, we begin with two beginner tutorials, both of which can be found at the list of App Inventor beginner tutorials webpage: http://appinventor.mit.edu/explore/beginner-tutorials-short.html

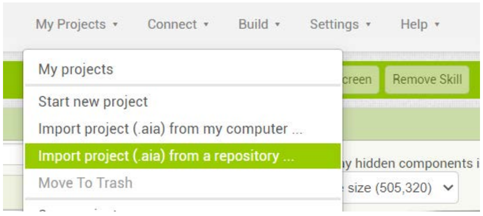
Tutorial 1: Codi Clicker! (~ 45 min)

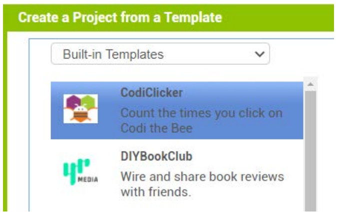
Goals

1. Learn how to navigate between the **Designer and Blocks** views in App Inventor
2. Learn the order of block execution and **how blocks interact**.
3. Learn how to use **global variables**.

The accompanying pdf for this tutorial can be found here. This tutorial is a modified version of the beginner App Inventor tutorial *Hello Codi!* (available here) where a button with a picture of Codi the bee makes a buzzing noise when you click on it. *Codi Clicker!* adds a label which counts the number of times Codi was clicked, resembling a very simple Clicker game. This way, we introduce students to variables.

Because of the media files (specifically an image and a sound file), there are two methods of starting this tutorial. The first and recommended method is importing the project from a repository. To do so, click on My Projects in the top bar, clicking “Import project (.aia) from a repository ...”, and selecting the *CodiClicker* template in the resulting pop-up window.





Alternatively, students can download a starter file [here](), and select “Import project (.aia) from my computer” instead. They would need to be able to find the downloaded .aia file and upload it in the new pop-up window.

Since this is the first time students are exposed to the App Inventor interface, the first thing you want to explain is the difference between the **Design view** and **Blocks view**. The Design view is for designing the user interface and importing all the components the student will want to fuse in their project. The Blocks view is where the block-based programming screen is.

Then, familiarize the students with the idea of **Visible Components** and **Non-Visible Components** when you drag the **Button** and **Player** components respectively for the tutorial.

When you get to the point of the tutorial where you place the “when Button1.Click” event block, make sure that students have a strong grasp of the concept of “events.”

What are events?

In App Inventor, most yellow blocks which start with “when” (like the “when Button1.click” block) are considered event blocks. Simply put, these blocks wait until something happens (the event). When that thing happens (e.g., the app-user clicks on the button), the blocks inside the yellow event block will run in order from top to bottom.

Once you get to the point where we are incrementing a global variable, we recommend you pause to explain how variables work. You can think of a variable as a container that holds a value that changes while you're using the app. You can read more about App Inventor variables here and here.

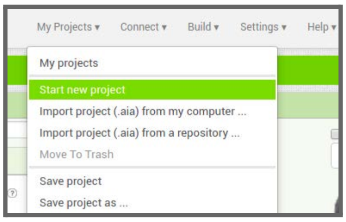
Tutorial 2: Talk With Me - Part 1 (~ 25 min)

Goals

1. Learn how to use the **TextToSpeech** component.
2. Learn how to use the **TextBox** component.

Similar to the *Codi Clicker!* Tutorial, the *Talk With Me* tutorial is a modified version of the beginner App Inventor tutorial *TalkToMe*, now introducing if statements, conditionals, and (in Part 2) -- a simple rule-based conversational agent. The link to the pdf tutorial is here.

To begin this tutorial, students should create a new project by selecting *My Projects* in the top toolbar and selecting *Start New Project* from the dropdown. Although the name of this new project can be anything the students would like, you can suggest "TalkWithMe."



The first part of this tutorial is very similar to the original *TalkToMe* tutorial on the Beginner App Inventor Tutorials website, with one main difference; we removed the accelerometer from the tutorial. Although this feature is fun and interactive, the core feature of the original *TalkToMe* tutorial we want to highlight is the **TextToSpeech** component.

By the end of Part 1, (everything up to the section **Part 2: Add an If Statement**) the students should have a screen with a textbox and a button. When the button is clicked, the **TextToSpeech** component will read the text in the textbox out loud.

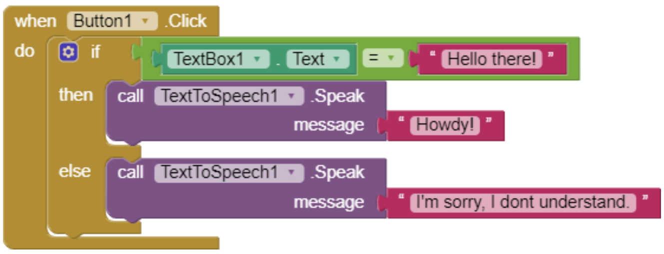
Tutorial 2: Talk With Me - Part 2 (Rule-Based Conversational AI) (~ 20 min)

Goals

1. Learn how to use the **if-then-else** block.
2. Understand **Rule-Based Conversational AI**.

Once they have finished Part 1 of the modified tutorial tutorial, we will introduce the idea of having the app respond to the text you enter in the textbox rather than reading it. By the end of this tutorial, students will have developed a very simple conversational agent!

The basic plan will be that when the user presses the button, the app will run a series of if-statement checks. If the text in the textbox reads “Hello there!”, the TextToSpeech component will respond with “Howdy!” If you type anything else, the TextToSpeech component will respond with “Sorry, I don’t understand.” Below is an example implementation.



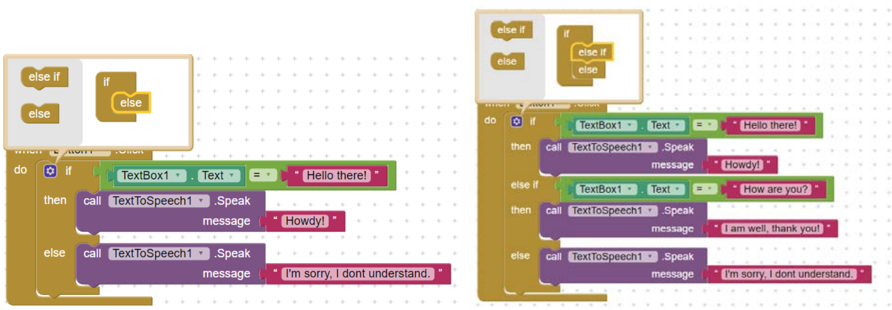
This logic actually outlines a very simple “chatbot” known as a **Rule-Based Conversational Agent**, where the response or action of the “agent” (aka code or robot) is determined strictly by rules defined by the programmer.

Rule-Based Conversational Agent

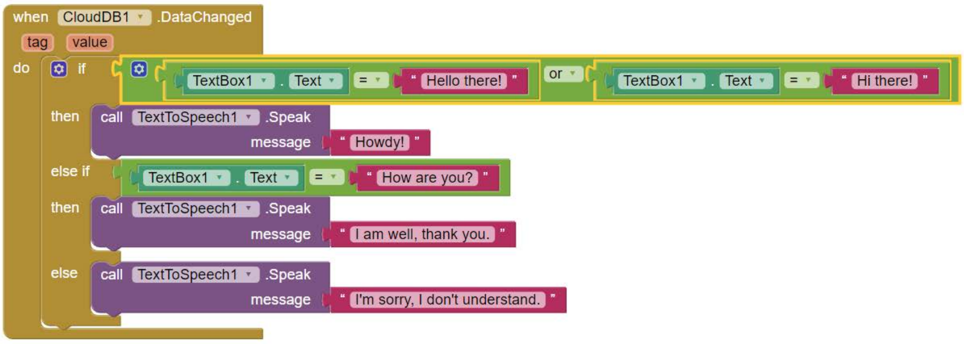
To begin with, conversational AI is any artificial intelligence which is capable of interpreting and responding to human language. Rule-based conversational AI is artificial intelligence which follows strict rules along the lines of “*If the user says X, then the AI says Y*”, to mimic human intelligence.

Once this simple if-else statement concept has been introduced and implemented, let the students play around with the app. Inevitably, some students will have trouble getting their conversational agent to “understand,” likely because of a typo, case-sensitivity, or a missing punctuation mark.

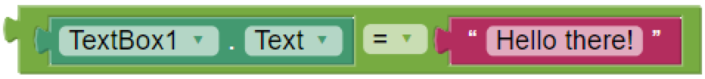
After giving them time to experiment, show the students how to append more conditionals to the if statement by using the gear box and dragging an **else if** and add a check for the phrase “How are you?” Below is an illustration of adding a new conditional:



Now, the students should get some more time to experiment and add new conditionals, gradually making their conversational agent more sophisticated. For some students, introducing them to the “or” block in the logic drawer would be helpful and relevant.



Some students may also find the block that converts text blocks to upper case / lower case helpful for addressing case-sensitivity. For example, we might use this in the conditional for the “Hello there!” check to remove the case sensitivity like so:



Now that the tutorial is complete and students had ample time to develop their custom “chat bot”, start a discussion about the shortcomings of conversational agents designed in this way. The most prominent issue is that if the user mistypes or says something similar in meaning to “Hello there!” that the developer had not prepared for, the device will always respond with “Sorry, I don’t understand.” One potential, though tedious, solution would be to use many nested OR statements. Plant the seed for the next day’s introduction to artificial intelligence by mentioning: “If only there was a way that the computer could generalize that all greetings (‘Hello’, ‘Hi’, ‘Hey’, etc.) mean the same thing.”

Introduce Alexa-Enabled Devices (~20 min)

The final task of the first day is to distribute Alexa-enabled devices to the students and give them time to interact with them using Alexa's default programming. If your organization does not have physical devices to hand out, students can download the free Amazon Alexa mobile app, which comes with an Alexa simulator that students can use as well.

For families that are reluctant to bring an Alexa-enabled device into their home or otherwise cannot get access to a physical Alexa-enabled device, direct them to Echosim.io, an online Alexa simulator.

Day 2 - Introduction to Alexa Skills

The second day of the workshop introduces students to Artificial Intelligence, AI Ethics, and finally the basics of creating their first Alexa skill using the **Alexa Interface**, an instance of MIT App Inventor designed for making Alexa Skills. This builds on the students' experience in Day 1 now that they are comfortable with the App Inventor programming interface and have gotten familiar with rule-based conversational agents.

Discuss: What is Artificial Intelligence? (~30 min)

We begin the day with an interactive activity. In this portion of the workshop, you will be guiding students to come up with a method of classifying what is or is not a device that uses **Artificial Intelligence (AI)**. The slides used during the workshop can be found here.

> **Artificial Intelligence**
>
> One definition for **artificial intelligence** is the development of computer systems which can perform tasks that ordinarily require human intelligence. Examples of such tasks include image classification (e.g., classifying a picture as a cat or a dog image), decision-making (e.g., deciding which move will lead to winning the game), and speech recognition.

The first part of this activity involves describing the Big 5 AI Ideas listed in this study. The 5 AI ideas are qualities that many people believe devices that employ AI share: Perception, Representation and Reasoning, Learning, Natural Interaction, and Societal Impact. This poster provides a neat summary of the ideas presented in the study.

The second part of this activity will show the students several examples of devices that may or may not be considered "AI", such as a toaster, a thermostat or a social media feed, asking students whether students believe these items involve AI and why, and then encouraging students to create their own definition of AI.

Discuss: AI Ethics (~30 min)

The slides used in the workshop for this discussion can be found here. This discussion can vary from teacher to teacher. For this workshop, the plan is to begin with a clip of Google Duplex, an advanced AI capable of making a call to local businesses and scheduling appointments (https://youtu.be/D5VN56jQMWM?t=45). Ask questions on whether or not it is okay to use AI without telling the other person that they are not actually talking to a human.

Another point of discussion could be bias in training data and thus in the AI. For example, immigrants or people with speech impediments have difficulty using Conversational Agents. These agents were trained to convert clearly spoken English (or other supported language) into intents, and so the AI is biased to understand only clear English. To be a little more sensitive, the presentation illustrates training bias through a captioning system trained on male speakers which would not perform very well when it encounters a female speaker for the first time.

Additionally, the topic of deep fakes is becoming increasingly relevant with the recent advancements of AI. Here is a cool video demonstrating the creation of deep fake video in real time, all based on a single image of the person you want to fake: (https://youtu.be/mUfJOQKdtAk?t=6)

The direction and details of this discussion can vary from student group to student group, so scaffold this conversation. Ask open ended questions and promote student interaction! Some questions are provided in the presentation, but some example questions are also provided in this document as well.

Example Questions

Here are a couple of questions taken from the slides linked above that could spark some discussion with the students. Depending on how engaged the students might get with a particular question, you don't need to use them all.

1. What jobs do you think that AI **cannot** do better than people?
2. What jobs do you think that AI **can** do better than people?
3. What do you think might happen to people with these jobs?
4. If you had a really good AI that can write about anything, what would you have it write about to make someone feel better?
5. What would you have it write about to make someone feel sadder?

Tutorial 3: Hello World Tutorial (~60 min)

Goals

1. Create your first **Alexa Skill.**
2. Understand what an **Invocation Name** is.
3. Understand what an **Intent** is.
4. Understand what an **Utterance** is.

The pdf for this tutorial can be found here. Now that students understand broadly what AI is, the students will finally develop their first Alexa Skill through this tutorial.

This tutorial will acquaint the students with the steps and vocabulary involved with creating an Alexa skill by making a simple skill that says hello or good-bye when the user says a greeting or farewell respectively.

The first concept that the student should understand is that every custom Alexa skill they create must have an **Invocation Name**. In short, an invocation name for your Alexa skill is analogous to an app name for a smartphone. Below is a description of the role that an invocation plays in an Alexa Skill.

What is an Invocation Name?

Just like how every mobile app needs to have a name, so does our custom Alexa Skill. An **Invocation Name** is just the name of the Skill that we are making, and is used to "invoke" our skill. The structure of any command you will tell Alexa is:

"Alexa, tell <Invocation Name> to …"

The invocation name is what will help Alexa tell which Skill it needs to use, so make sure that every skill you make has a *unique* name.

For example, if we decided to make the Invocation name of a custom skill "Codi bee" we would say:

"Alexa, tell Codi bee to do something"

Note: The invocation name needs to be at least two (2) words long, but avoid making it a full sentence, since you will be saying the name a lot.

The next concepts that students should familiarize themselves with are intents and intent phrases. An **intent** can be thought of as a command that your skill should be prepared to respond to. **Intent phrases** are different sentences or phrases which all share the same intent. For example, when you say "Hello", "Howdy", or "Hi", they are all greetings. These three sentences are the intent phrases, and the intent behind all these phrases is the same: greetings! The AI will then learn the pattern from your provided intent phrases and be able to guess what other phrases might also have the same intent.

Finally, an **utterance** is the specific sentence that the user says to trigger a particular intent. For example, if the user says “Alexa, tell Codi Bee to say hello”, “Codi Bee” is the invocation name and “say hello” is the utterance which falls under our greetings intent from earlier. The general format of any command you tell Alexa will fall under this pattern:

“Alexa, tell <Invocation Name> to <Utterance>”

Be sure to explain these concepts as you walk the students through the tutorial. In the next day, we will begin to explain some of the AI that Alexa employs in order to understand that “Say hello” has the same intention as “Howdy” or “Hi.”

Day 3 - Machine Learning and Slots

The third day will begin a dive into the role that Machine Learning plays in an Alexa skill. Additionally, we will show students how to extract additional information from a user's utterance by using **slots**.

Discuss: Machine Learning and Transfer Learning Slides (~30 min)

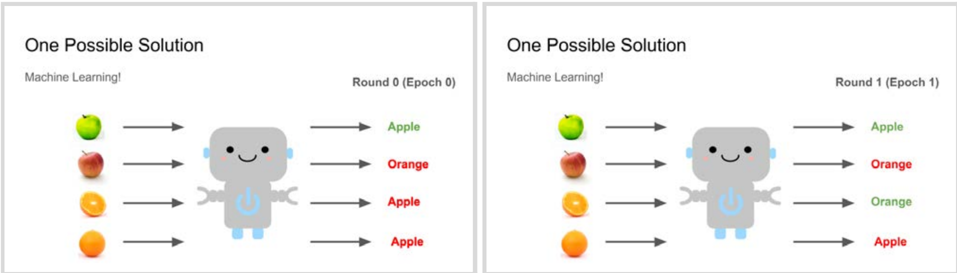
Goals

1. Understand **Machine Learning**
2. Understand **Natural Language Processing**.
3. Understand **Transfer Learning**

The first activity of the third day is a presentation on the role of machine learning in our conversational agent. The presentation can be found here. First, review with the students the basic components of an Alexa Skill they've learned from the previous tutorial: invocation names, intents, and utterances. Then, ask students to compare and contrast this simple Alexa Skill with their rule-based conversational agent. Ideally, a student will eventually say that Alexa was able to recognize words and phrases that they hadn't specifically listed as an intent and respond how you might expect. If they don't get to this point on their own, attempt to guide their attention to this key contrast.

The reason for this ability to generalize lots of phrases as an intent when you only give it a few examples is a direct result of Machine Learning. We've created a slideshow for a discussion on **Machine Learning, Natural Language Processing (NLP)** and **Transfer Learning**.

First and foremost is a very high level definition of Machine Learning. In the slides, we provide an example of image classification, since images are more intuitive for students. The robot must classify a given image as an Apple or an Orange. This robot has a couple of rounds of training known as **epochs** to demonstrate that this machine is learning and getting better at telling apart apples from oranges. This task of classifying pictures as an apple or orange is analogous to classifying an utterance as a “greeting” or “farewell” intent for our Alexa Skill.



> **Machine Learning (ML)**
>
> **Machine Learning** is a subset of Artificial Intelligence. Machine learning models refers to instances of the machine learning algorithm. You can think of the model as a “robot brain” for the sake of high level explanation. In order to improve the ML model and make it smarter, you “**train**” the model by showing it a lot of data and examples of the answer. Later in this workshop, we will go much more in depth into some of the math behind the learning.

At a high level, Alexa uses NLP to be able to generalize the sample phrases you provide for flexibility, but only if you can provide multiple phrases for the same intent. You can relate this to the image classification example. Instead of apples, it’s different sentences that mean Hello, and instead of oranges, it’s different sentences that mean good-bye. The model will learn which one means HelloIntent, and the other is the GoodByeIntent.

Natural Language Processing (NLP)

Natural Language Processing (NLP) is broadly defined as a subfield of artificial intelligence which focuses on computers' abilities to understand and interact with human (natural) languages. Tasks that require NLP include translating phrases between different languages, understanding that different phrases carry the same intention (e.g., "hello" is similar in meaning to "hi"), and categorizing words as specific parts of speech (e.g., "dog" is a noun). As you can guess, conversational agents like Amazon Alexa and Google Home employ NLP.

Once you explain that Amazon is using AI to generalize the meaning of the phrases for your intent, you may choose to mention that this process of training AI is normally very slow and time is money. Amazon wants to train your model to be accurate as quickly as possible. Thus, programmers typically use **Transfer Learning**, where they don't actually start learning with an "empty brain." Rather, they start with a "brain" that *already* knows English very well, and then fine tune this head-start. Thanks to this strategy, the end result is both smarter than if we started with an "empty brain" and it learned much faster!

Think of it like teaching a newborn baby to analyze Shakespeare vs teaching a college student to analyze Shakespeare. It would take much longer to teach the newborn baby English first and then have it analyze Shakespeare. You can completely skip the "learn English" step for the college student, who can **transfer** their understanding of English to the new specialized task of analyzing Shakespeare. We've created a slideshow to help teach the broader concepts of machine learning and transfer learning, which you can find here.

Transfer Learning

Early researchers in the field of Machine Learning would start to train the model from a blank slate.

A more recent practice that developers are doing is to use transfer learning. Transfer learning allows you to use an existing model trained for some task, and "transfer" that knowledge to a new task. This can save time and resources especially for smaller businesses or research groups.

Tutorial 4: My Calculator Tutorial (~60 min)

Goals

1. Learn how to use **Slots** to get information for the Alexa Skill.
2. Learn about blocks in the math drawer.

The pdf for the *My Calculator* tutorial can be found here. This tutorial will be the students' introduction to what Amazon Alexa Skills refer to as **Slots**. Slots are basically blanks, and the Alexa skill will figure out what words the user said belong in the blanks you've set. They are very similar to variables. It may be worth taking the time to walk the students through a very simple example, like telling students to parrot back a response like:

MIT APP INVENTOR

Slots Activity Example

When I say "My name is ___," you should say "Hello ___."

Teacher: My name is <Your Name>.
Students: "Hello <Your Name>.

Teacher: "My name is Luke Skywalker.
Students: "Hello Luke Skywalker.

Teacher: "My name is not my name."
Students: "Hello not my name.

The underlined words would be considered "name slots".

After this activity, you can explain that slots are the blanks and you can actually use this to do math. For example, if you have two "number slots" in the same sentence, like "What is __ plus __", then we can program Alexa to add the numbers in the slots and respond with the answer (e.g., User: "What's two plus seven?", Alexa: "Two plus seven is nine"). After this, you can begin the My Calculator tutorial.

Discuss: ML Behind the Scenes and LSTM Slides (~30 min)

After the *My Calculator* tutorial, we return to the topic of Machine Learning. The slides for this discussion can be found here. This time, we will be taking a more detailed look into the learning process. The goal here is to demystify the "learning" of AI. If nothing else, students should at least take away that AI is a very complex function that converts number inputs into number outputs.

Part 1 - What is Machine Learning?

Officially, there is a difference between AI and Machine Learning. AI is a term for any algorithm that displays human-like decision making abilities. Machine learning algorithms are a form of AI which makes decisions after having trained or "learned" from a large amount of training data. Below is a more detailed description of the difference, but to the layman, AI and Machine Learning are interchangeable and for the sake of teaching students, they are treated as such in the presentations.

> **AI vs Machine Learning?**
>
> Any computer algorithm which can make decisions intelligently can be described as AI. By this definition, Machine Learning is a specific kind of AI. Machine learning refers **only** to algorithms which can make decisions intelligently by learning rules or patterns through a process called *training*.
>
> For example, our original rule-based conversational agent can be described as artificial intelligence, but **not** as machine learning. On the other hand, the neural network that is shown in the slideshow is a form of artificial intelligence **and** machine learning, because it changes its weights over time and thus "learns"

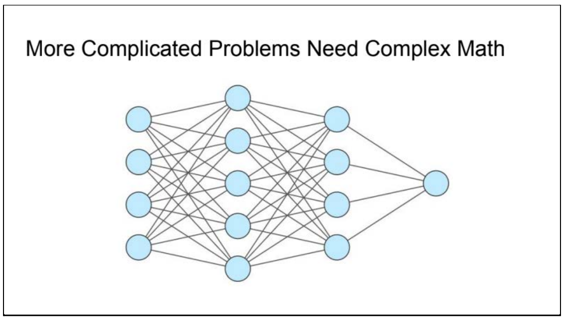
In the slides, we list three phases and challenges of training and developing an AI: encoding data, processing data, and improving the algorithm.

The first challenge is to encode your inputs, listed in the slides as "How to Make Things Numbers?" Since computers like working with numerical inputs, we need to convert real life data like pictures and words into numbers somehow. For example, images can be treated as a list of numerical pixel data. Text can be converted into a list of numbers through one-hot encoding, or more simply by assigning a unique number to each letter of the alphabet.

The second challenge is to process the data, listed in the slides as "How to Get Numbers Out?" To illustrate this as simply as possible, we decided to describe a single neuron in a neural network as the method of getting numbers out (ignoring activation functions, which is beyond the scope of this high level discussion). The idea here is you get students to understand that each neuron connection has a **weight** associated with it, which you multiply the inputs by and add to the output neuron. The number of neurons in each layer can be changed as well to allow for lists of data (like the list of pixel data or list of text values).

The third and final challenge is to improve the algorithm, listed as “How to Learn?” To illustrate this process, we describe “nudging” the weight in the direction of the correct answer. The reason for nudging instead of assigning the weight to the known correct answer is because (1) more complicated brains will be difficult to find the “correct answer” to assign, and (2) when learning over many data examples, the “correct” answer for one data sample might not be the “best” answer overall.

Finally, we bring up the important point that data gathered in real life is often too complex for a single neuron. As such, we layer these neurons into a large web of neurons called a **neural network**, inspired in part by brains. The more complicated the problem, the more complicated the neural network will be. This then transitions us to the second part of the presentation.



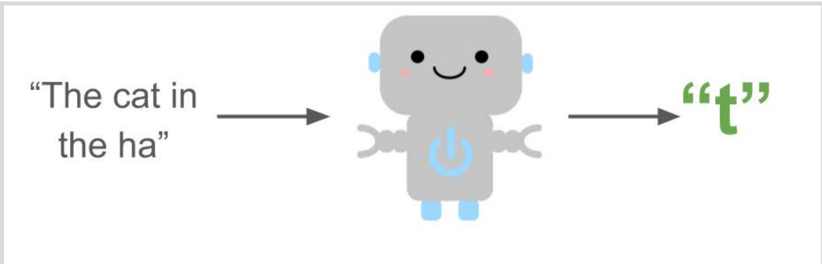
Part 2 - LSTM Models

In the above presentation, we describe two kinds of “robot brains.” The first is a **Neural Network**, which consists of layers of nodes and weights. Neural networks can do pretty well at a wide range of tasks, like deciding if a picture is of an apple or orange. However, there are other kinds of “robot brains” which are better at certain tasks.

One of these alternate “robot brains” is an **LSTM model**. This robot brain was wired in a way that it can “remember” old inputs, which makes it really good at tasks which have a time-dependent component like grammar. For example, if you didn’t remember that the subject of the sentence is “The dog,” you might think the verb could be “meows.” If you did remember, then you might instead suggest the verb be “barks.” The very basic LSTM models built-in to the Alexa Interface will take **seed** text and predict what the next letter.

LSTM Models

Long Short Term Memory (LSTM) models are a type of robot brain that has been wired to be able to remember old inputs and make smarter decisions based on its memory. Unlike the jack-of-all-trades that Neural Networks are, these LSTM models are especially good at time-dependent tasks, like predicting stock prices, reading music, or generating text! Each of these tasks benefit from being able to remember old data. Maybe yesterday’s stock prices can help me predict todays? The previous music notes can tell me what key the song is in. If you remember that the subject is plural, you can choose a verb that agrees.



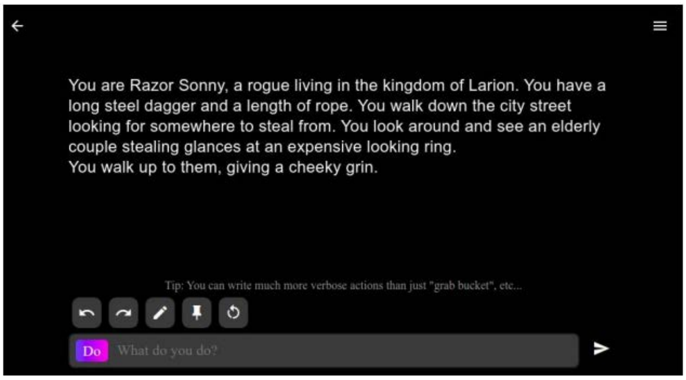
Despite the depth provided in the presentation here, it is up to you as the teacher to adjust the depth of this discussion to be appropriate to your students.

Discuss: Meet AI Dungeon (~20 min)

After discussing the Machine Learning and LSTM Models, if time permits, we highly recommend leading an activity with AI Dungeon, a free online demo of the GPT-2 language model. AI Dungeon is an online text-based roleplaying game that uses a language model trained on various genres, most notably fantasy genre.

The demo gives you a description of your goal and the setting, and prompts the user to enter an action they wish their character to perform. The language model then generates a handful of sentences describing the outcome of your actions, and prompts you for the next action. If you are familiar with the popular tabletop role-playing game Dungeons & Dragons, the language model takes the place of the Dungeon Master, hence the namesake of AI Dungeon.

To make this interactive, take suggestions from the students as to what the background of their character would be, the name of the character, and a handful of actions. During the activity, highlight points where the text-generation AI is able to remember key details from very early on in the story, such as the character's name, which was mentioned at the very beginning. This should highlight the role that memory plays in generating a realistic story.



An example of the starting text for AI Dungeon

Amazon Future Engineer Workshop: Teacher's Guide - 24

Day 4 - CloudDB and Communicating with an App

The goal of this day is to teach students how to create an Alexa Skill that can communicate with an app made using App Inventor by using the CloudDB component.

Although we won't be going through this CloudDB Chat App Tutorial during the workshops, it is a great resource for some additional practice using the CloudDB component as a precursor to this tutorial if time allows.

In short, the **CloudDB** component is a component of App Inventor which allows devices to connect to an MIT-owned database. Using the CloudDB component, devices can change values on CloudDB, read values from CloudDB with a given tag, and even detect when **another** device changes a value on the CloudDB.

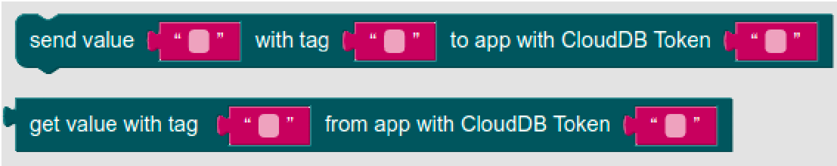
Tutorial 5: Read My Text Tutorial (~90 min)

Goals

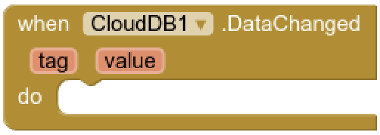
1. Learn how to use **CloudDB** to communicate between skills and apps.

The link to this *Read My Text* tutorial can be found here. Before we begin the tutorial, it is important that you explain to your students (1) what CloudDB is and (2) the role that CloudDB will play in creating an interactive Alexa Skills. To help with visualizations, we prepared a presentation found here. In order for the Alexa skill to interact with apps on a tablet or phone, we will be using CloudDB as a middle-man. Alexa-enabled devices cannot communicate directly with the app itself, so we will have it communicate with the CloudDB server instead.

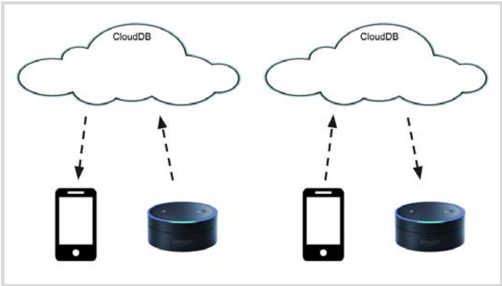
Unlike the normal interface with App Inventor where you would need to drag the CloudDB component onto the phone in order to use CloudDB, the Alexa Interface has two built-in CloudDB blocks for sending and receiving data from the CloudDB Server. Specifically, the blocks in the Voice Drawer look like:



These are the **only** blocks that Alexa can use to work with data on CloudDB. Whenever the Alexa Skill uses the ***send value __ with tag __*** block, this counts as changing data on the CloudDB. This allows for the developer to take advantage of the App Inventor's CloudDB event block which detects whenever data changes on the server. This block looks like this:



This is how we can use CloudDB to allow the **Alexa Skill to talk to App Inventor Apps**. Next, we want to allow the app to talk to the Alexa Skill. Unfortunately, we can't have Alexa talk based on DataChanged like we do with the App. This is due to privacy concerns on Amazon's end. (Imagine how freaked out you would be if a start-up could use their app to talk to you through your Alexa.) Thus, our work around is to use the ***get value with tag __*** block to read information that the app might have left for App Inventor. This means that in order for Alexa to read something that the app told it to, the user must ***start a conversation with Alexa first***.



Once you have explained the role which CloudDB is going to play in creating our interactive Alexa Skills, you can begin the tutorial.

Discuss: Brainstorm Ideas for Individual Projects (~40 min)

The presentation used during the brainstorming activity can be found here. During this period, help students come up with their own ideas for conversational agents and apps using Amazon Alexa and App Inventor. We recommend providing a theme to inspire students, and provide some example projects.

Some Example Themes

- Memory loss
- Blindness
- Movement difficulty
- Mental health
- Muteness

- Disability awareness
- Recycling
- Composting
- Climate change

- Environmental awareness
- Energy management
- Pollution
- Learning impairment

Discuss: Begin Project Outline (~20 min)

Due to the nature of remote learning, the students worked independently for their final projects. We also made it clear that we did not expect a finished project at the end of Day 5, since we understand that there is not a lot of time or man-power to expect that.

Once students have chosen the project that they would like, we divided the class into Zoom breakout rooms with a handful of students to 2 teaching assistants (TAs). The TAs would then ask each student to provide an example conversation with the skill they wish to develop. We found that asking for different conversations with their skill is better than simply asking students "what their skill will do" because it helps the students decide what intents and utterances they need for their Skill. Additionally, it is up to you and your TAs to provide discretion as to what is or is not a reasonable project and push them to avoid overly ambitious projects.

Day 5 - Individual Projects

Independent Work (~90 min)

Since this workshop is currently designed for remote learning, we plan on making sure everyone remains on the call, working individually, and if students have any questions, we would enter breakout rooms. Of course, if this workshop were conducted in person, this would be much simpler and they could be working in groups.

We advise that you create a Google Slides document (example slides), assign a number of slides on the slideshow to each student/group, and have them edit their respective slides. The slides assigned to each team should explicitly list or bullet point information which you would like the students to add to their presentation, such as the purpose or an example conversation a user may have with the skill.

Additionally, we highly encourage developing projects with a mobile component in addition to the Alexa Skill. Multimodal interaction (that is, having communication between an app on your phone and your Alexa Skill) can inspire many interesting projects!

An important limitation in programming Alexa Skills is that you cannot press a button on a phone or other external device and have an Alexa-enabled device to talk without having spoken the “Alexa” key word. For example, you cannot make a TalkToMe app where Alexa would read the text immediately when you press the button. In order for Alexa to speak, ***you must speak to Alexa first***. According to Amazon, this is due to privacy concerns. This kind of feature could allow for a developer to have Alexa “ask a question” without you having spoken to it and listen to your surroundings. In short, the app cannot talk to the Alexa Skill directly. It can only talk to CloudDB and have the Alexa Skill read the information from CloudDB **after you say an intent**.

Tips for Leading the Independent Work Time

As mentioned before, the original workshops were delivered through remote instruction through Zoom. These tips will be mainly for further remote instruction, but many of the tips can generalize to different methods of instruction.

In short, divide students into groups of individual projects. The instructors should remain in a group, going around and speaking with each student in the group at least twice. On the first pass, you might ask students what they want to work on and what an example conversation with the skill would be. Suggest a couple intents they might want to create before moving on to the next student. On the second pass, provide students with information about some blocks and some logic that the students would find relevant. Finally, on subsequent passes, ask students what they are working on and if they have any questions or need any help. Repeat this final step until the time is up for final projects.

Tip 1 - Divide Students into Groups Based on Project Themes.

In the original workshop, we divided into breakout rooms with roughly two instructors to five or six students. This allows for more open communication with the students.

Tip 2 - Give Each Student 1-on-1 Time to Describe their Project

The second tip is to go around to each of the students and ask them one-by-one what they want their final project to be about, what problems they want to solve or what mechanics would be a part of their game. Continue to ask for details and do your best to avoid providing your own ideas for features of their skill. Your role should be to provide them the support to bring their idea to light. Once they begin implementing the Alexa Skill, you can start to suggest simplifications, but for now let them brainstorm features.

Tip 3 - Ask Each Student for an Example Conversation With their Skill

How do the students expect a human to interact with their skill and how do they expect the skill to respond? This question grounds students and clarifies the direction of their app. At this point, you should suggest one or more intents for the student to begin defining phrases for.

Tip 4 - On the Second Pass, Suggest to Each Student a Handful of Relevant Blocks

The fourth tip is, after you make the first round and get each student started creating their intents, suggest students some blocks that would be relevant to their intent. If the number of students in your group is large, you may want to suggest this on the first pass so that students are not idling for too long.

Tip 5 - Subsequent Passes, Ask for Progress, Not If They Need Help

Finally, subsequent passes should be about pushing students in the right direction as they work independently. Often, students are too embarrassed to answer when an instructor asks if “anyone needs any help,” so some advice would be to instead ask students how they are progressing and what they are currently working on. Once students begin describing what they are working on, they will naturally point out parts of their app/skill which are causing them trouble. This discussion might also point out some flawed implementation, in which case you may choose to explain a different implementation of different features to accommodate the limited development time.

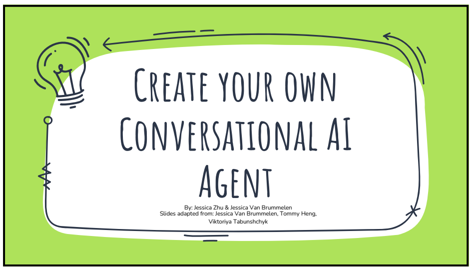
Share Your Skills (~20 min)

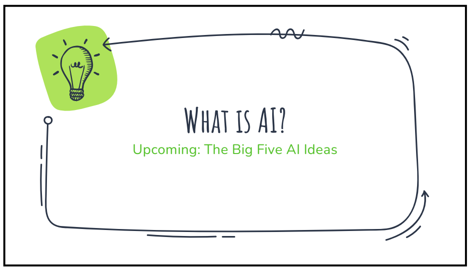
At the end of the day, each student/group should talk about what they were working on with a slide or two. Depending on the number of students, you may choose to either allot more time for the presentations, skip some groups, or only have volunteers present.

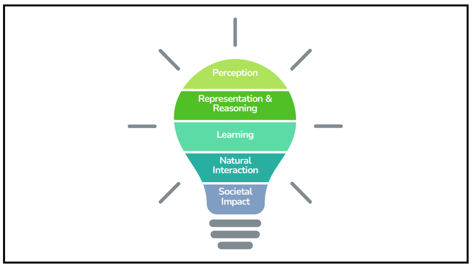
Appendix G

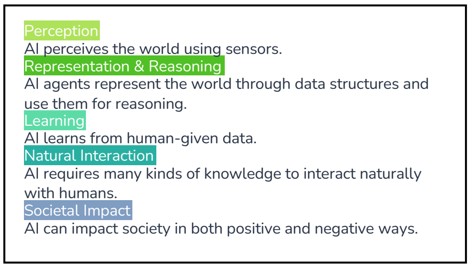
Convo Teaching Resources

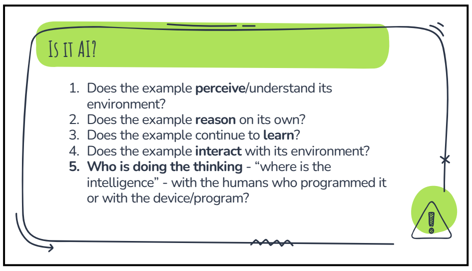
This appendix contains teaching resources for the Convo activity [190, 215], including an example presentation from the workshops and two example tutorials. The Convo repository is stored on GitHub [204].

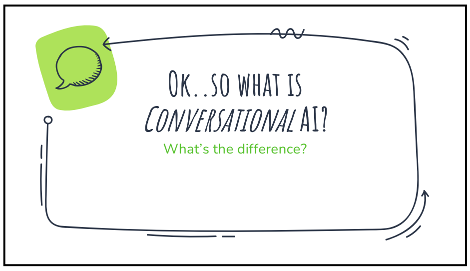

CREATE YOUR OWN CONVERSATIONAL AI AGENT
By: Jessica Zhu & Jessica Van Brummelen
Slides adapted from: Jessica Van Brummelen, Tommy Heng, Viktoriya Tabunshchyk

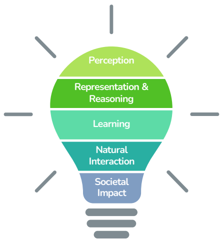

WHAT IS AI?
Upcoming: The Big Five AI Ideas


Perception
Representation & Reasoning
Learning
Natural Interaction
Societal Impact


Perception
AI perceives the world using sensors.
Representation & Reasoning
AI agents represent the world through data structures and use them for reasoning.
Learning
AI learns from human-given data.
Natural Interaction
AI requires many kinds of knowledge to interact naturally with humans.
Societal Impact
AI can impact society in both positive and negative ways.


IS IT AI?
1. Does the example perceive/understand its environment?
2. Does the example reason on its own?
3. Does the example continue to learn?
4. Does the example interact with its environment?
5. Who is doing the thinking - "where is the intelligence" - with the humans who programmed it or with the device/program?


OK..SO WHAT IS CONVERSATIONAL AI?
What's the difference?



#1: Perception

Conversational AI agents perceive the world through user-given speech and text.

#2: Representation & Reasoning

Conversational AI agents represent the world through **intents** and **entities**.

They reason by trying to match what a user says to known commands.

#3: Learning

Conversational AI agents mainly learn from training data, but advanced versions can learn from conversations with users.

#4: Natural Interaction

Conversational AI agents must be able to understand what humans say and respond accordingly.

#5: Societal Impact

Conversational AI agents can help humans do simple tasks, answer questions, and much more!

Perception
Conversational AI perceives the world through text and speech.
Representation & Reasoning
Conversational AI represents the world through intents and entities and use them to interpret what a human means.
Learning
Conversational AI learns from human-given training data and conversations.
Natural Interaction
Conversational AI understands and can respond to human language.
Societal Impact
Conversational AI can help humans answer questions, perform tasks, and more!

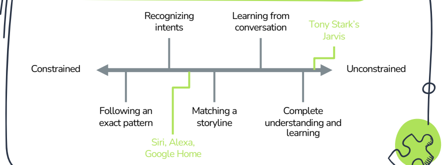
Convo is a conversational programming agent.

Convo is a conversational programming agent.

Conversational: Natural Language Models

Constrained	Unconstrained
Human learns how AI communicates	AI learns how humans communicate
Has instruction manual	No instruction manual
Harder to learn	Easier to learn
Easier to create	Harder to create

Conversational: Natural Language Models



Instruction Manual

Greet
[Hello/Hi] my name is [Jessica].

Goodbye
[Goodbye/Bye/Farewell].

Hey!!
Hello my name is Jess.
Hi my name is Greg.
How are you doing?

Bye
Until next time!
See you later
Goodbye

Convo is a conversational programming agent.

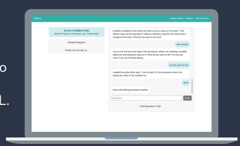
Programming Agent

- Can create programs
- Can execute programs
- Responds when you communicate to it

You can think of Convo as a chatbot that can help you program and can learn to recognize certain intents that you teach it.

How to Use Convo

Convo is a web app, so you can access it by going to a specific URL.



21

Next Week:

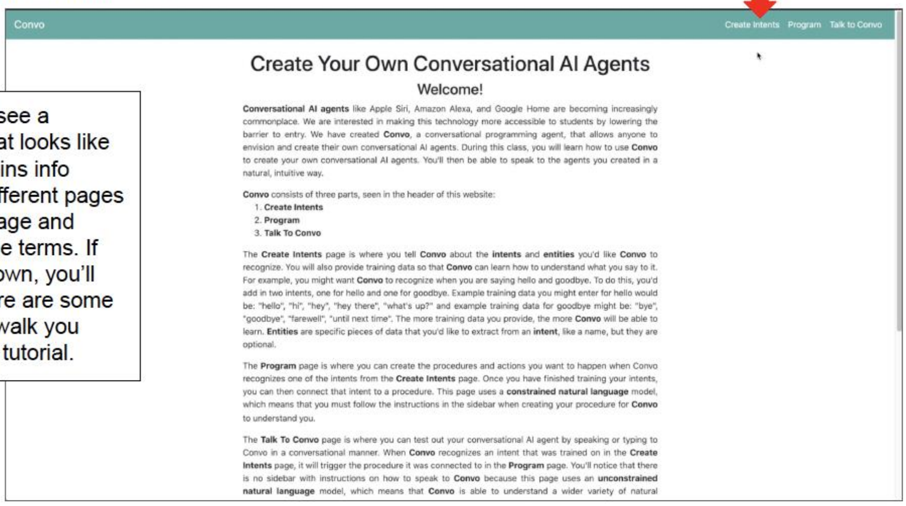
- Convo Starter Task
- Project Ideation
- Create with Convo!

Convo Tutorial Part 1

Let’s teach Convo how to greet us and say goodbye!

Step 1: Navigate to the URL
userstudy.appinventor.mit.edu

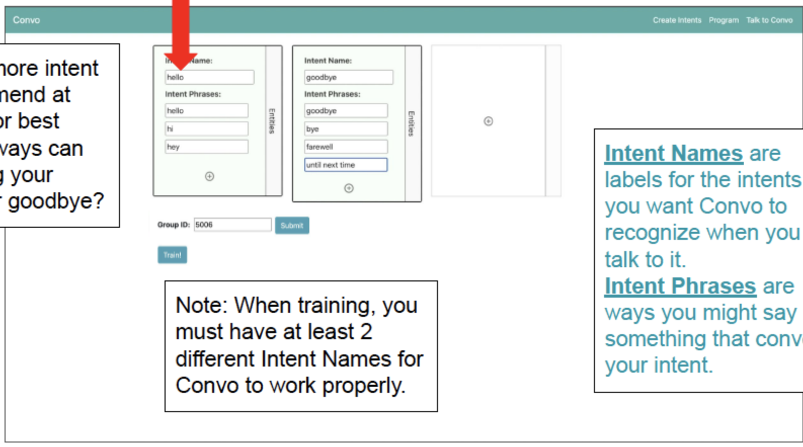
Step 2: Go to the Create Intents page



You should see a webpage that looks like this. It contains info about the different pages of the webpage and defines some terms. If you scroll down, you'll see that there are some videos that walk you through this tutorial.

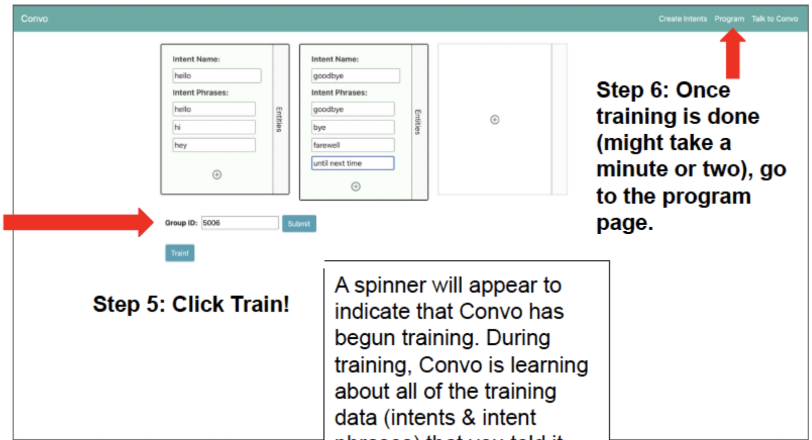
Step 3: Enter in the intents and intent phrases

Feel free to add in more intent phrases, we recommend at least 15 per intent for best results. How many ways can you think of showing your intent to say hello or goodbye?



Note: When training, you must have at least 2 different Intent Names for Convo to work properly.

Intent Names are labels for the intents you want Convo to recognize when you talk to it.
Intent Phrases are ways you might say something that conveys your intent.



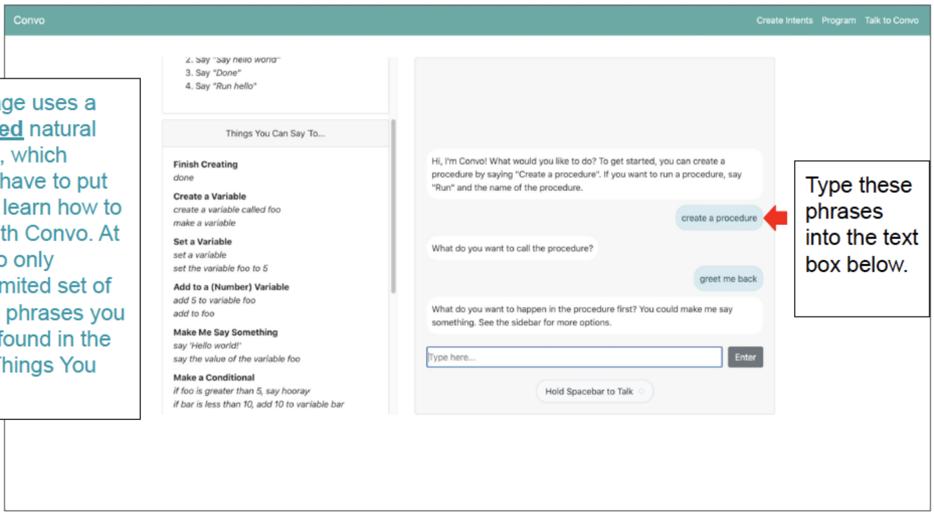
Step 4: Put in your Group ID (given to you in class) and click Submit.

Step 5: Click Train!

Step 6: Once training is done (might take a minute or two), go to the program page.

A spinner will appear to indicate that Convo has begun training. During training, Convo is learning about all of the training data (intents & intent phrases) that you told it about.

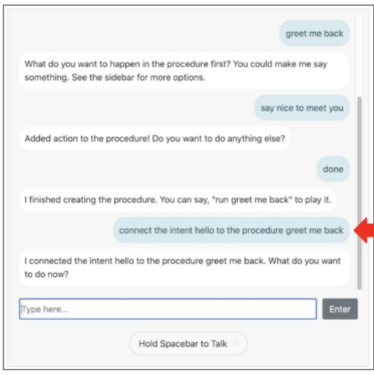
Step 7: Create the procedure you want Convo to do when you greet it.



The Program page uses a more **constrained** natural language model, which means that you have to put in some work to learn how to communicate with Convo. At any point, Convo only understands a limited set of commands. The phrases you can say can be found in the sidebar under 'Things You Can Say To…'

Type these phrases into the text box below.

Step 8: Finish adding actions to the procedure.



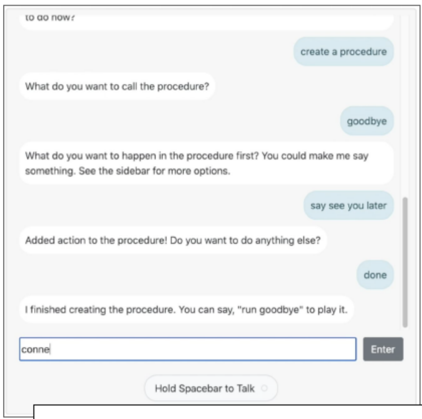
We start off by telling Convo that we want to create a procedure. Then, we name our procedure 'greet me back'. You can choose a different name if you wish. We want Convo to say 'nice to meet you' when it detects that we are saying hello. We finish by typing 'done'.

Step 9: Connect the procedure to the corresponding intent.

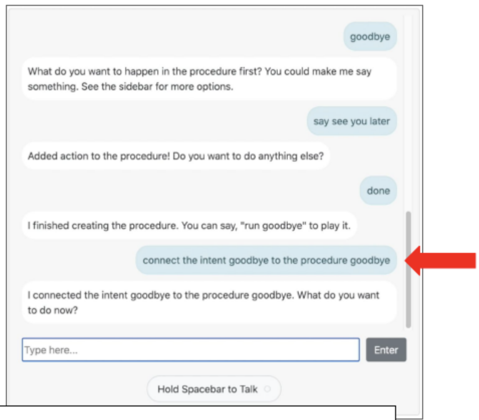
Connecting a procedure to an intent will make Convo run the procedure you just created whenever the connected intent is triggered on the Talk to Convo page.

You can choose to either type or talk to Convo. To talk, either hold the record button, or hold your spacebar (click outside of the chatbox first). When recording, the red light should turn on:

Step 10: Create the procedure you want Convo to do when you say goodbye.

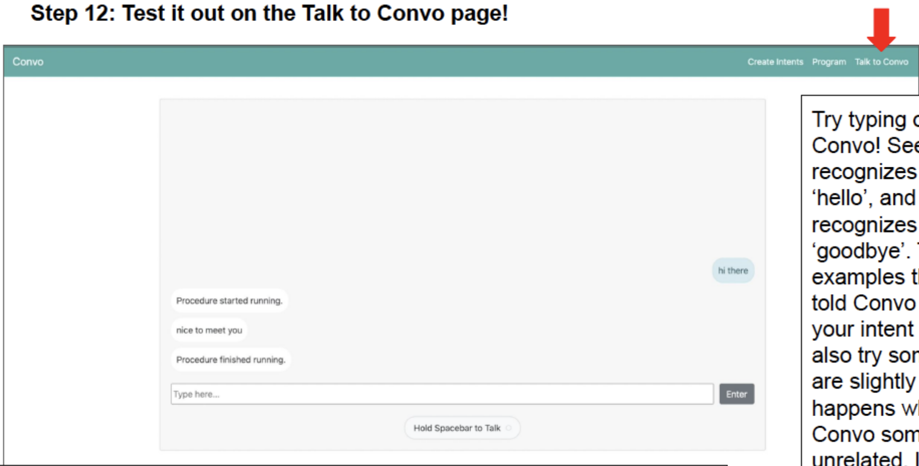


Step 11: Connect the goodbye procedure to the intent goodbye.



This is the same process as creating the procedure 'greet me back'. You can choose to have Convo say something different, or you can also add in more actions from the choices in the sidebar. Connecting the intent to the procedure executes (runs) this procedure (or set of actions) whenever the intent 'goodbye' is detected by Convo.

Step 12: Test it out on the Talk to Convo page!



Try typing or talking to Convo! See what Convo recognizes as the intent 'hello', and what it recognizes as the intent 'goodbye'. Try saying examples that you directly told Convo about through your intent phrases, but also try some things that are slightly different. What happens when you tell Convo something super unrelated, like "cats and dogs"?

The Talk to Convo page uses a more **unconstrained** natural language model, which means that there's no sidebar of acceptable commands. Convo will do its best to interpret what you say and match it to a known intent.

Convo Tutorial Part 2

Let's teach Convo to tell us the weather for a certain city!

Step 1: Navigate to the URL **userstudy.appinventor.mit.edu**

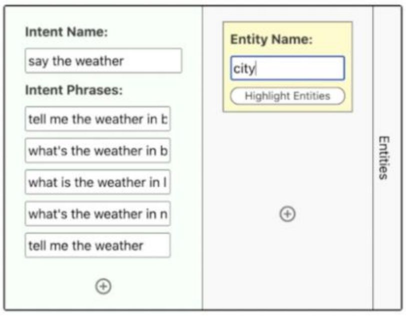
Step 2: Go to the Create Intents page.

You should see a webpage that looks like this. It contains info about the different pages of the webpage and defines some terms. If you scroll down, you'll see that there are some videos that walk you through this tutorial.

Step 5: Add an entity.

We want Convo to be able to extract the 'city' from what we say, so that it can give us the weather for that specific city. To do this, we need to tell Convo to look for cities, and we do this by creating an entity.

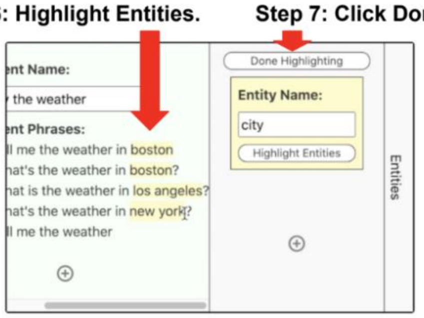
As with intent phrases, Convo will also be more accurate the more entity examples you give it. We recommend at least 15 unique entity values (in this case, we used Boston twice, which only counts as 1 unique entity value).



__Entities__ are pieces of information we want Convo to be able to extract from what we tell it.

You can have multiple entities per intent. You would need multiple entities if you are trying to extract multiple pieces of information, such as getting the *date* in addition to the *city* for a weather forecast. The current limit is 6 entities per intent.

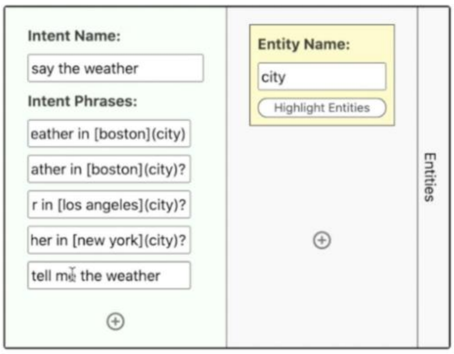
Step 6: Highlight Entities. **Step 7: Click Done Highlighting.**



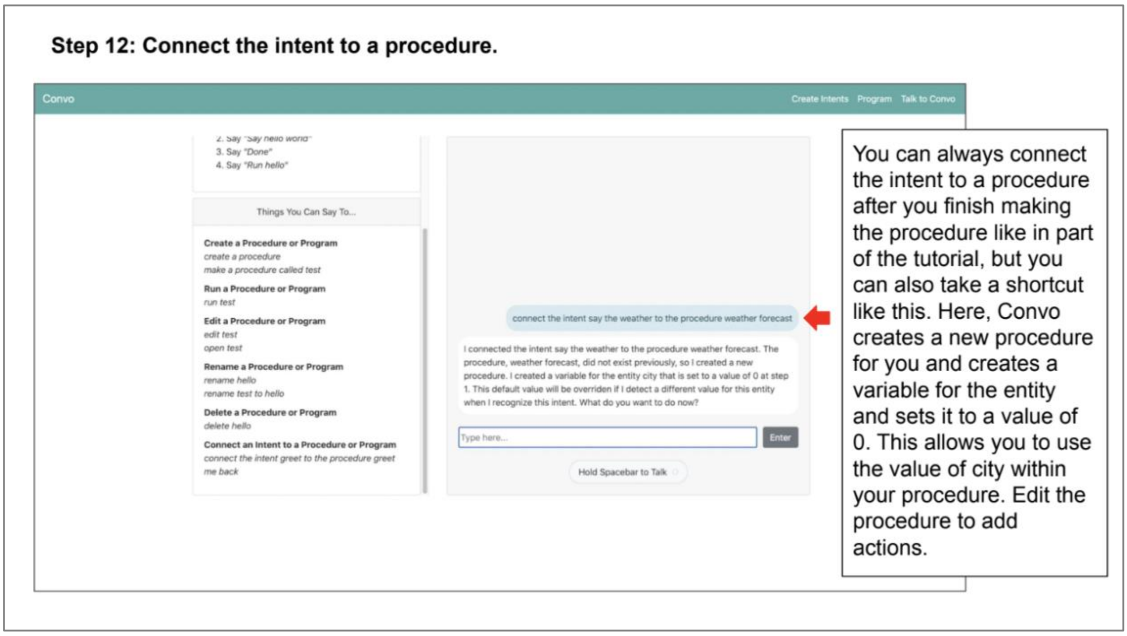
(Alternatively, you could directly enter in your intent phrases with the entity syntax, [*boston*](*city*), and you can skip the highlighting entirely.)

Simply drag your cursor over words that you'd want Convo to understand as a 'city'. Please be careful during this step since there's currently no undo button (at this step). If you do make a mistake, no worries. Simply click Done Highlighting and manually edit the Intent Phrases. The entity value should be in hard brackets [], followed by the entity name in parenthesis (), with no space in between.

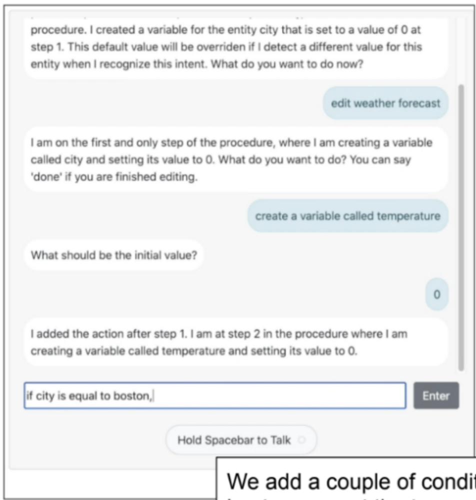
Step 8: Fix any mistakes.

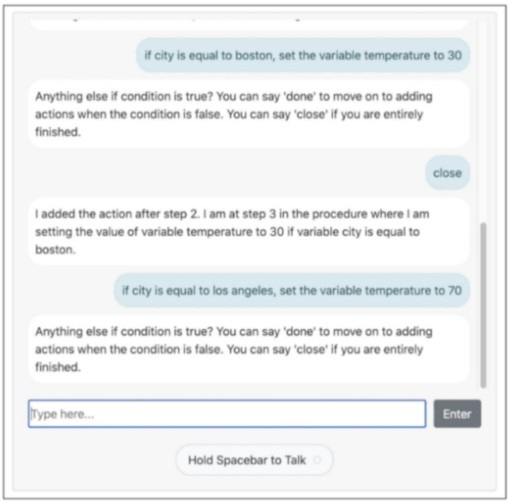



Step 11: Once training is done, go to the program page.
Convo
Create Intents Program Talk to Convo
Step 9: Put in your Group ID (given to you in class) and click Submit.
Step 10: Click Train!
A spinner will appear to indicate that Convo has begun training. During training, Convo is learning about all of the training data (intents, intent phrases, and entities) that you told it about.


Step 12: Connect the intent to a procedure.
Convo
Create Intents Program Talk to Convo
Things You Can Say To...
Create a Procedure or Program
Run a Procedure or Program
Edit a Procedure or Program
Rename a Procedure or Program
Delete a Procedure or Program
Connect an Intent to a Procedure or Program
connect the intent say the weather to the procedure weather forecast
Hold Spacebar to Talk
You can always connect the intent to a procedure after you finish making the procedure like in part of the tutorial, but you can also take a shortcut like this. Here, Convo creates a new procedure for you and creates a variable for the entity and sets it to a value of 0. This allows you to use the value of city within your procedure. Edit the procedure to add actions.

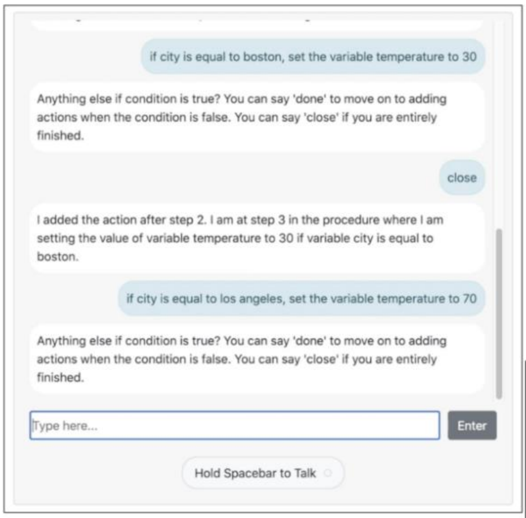
Step 13: Finish adding actions to the procedure.





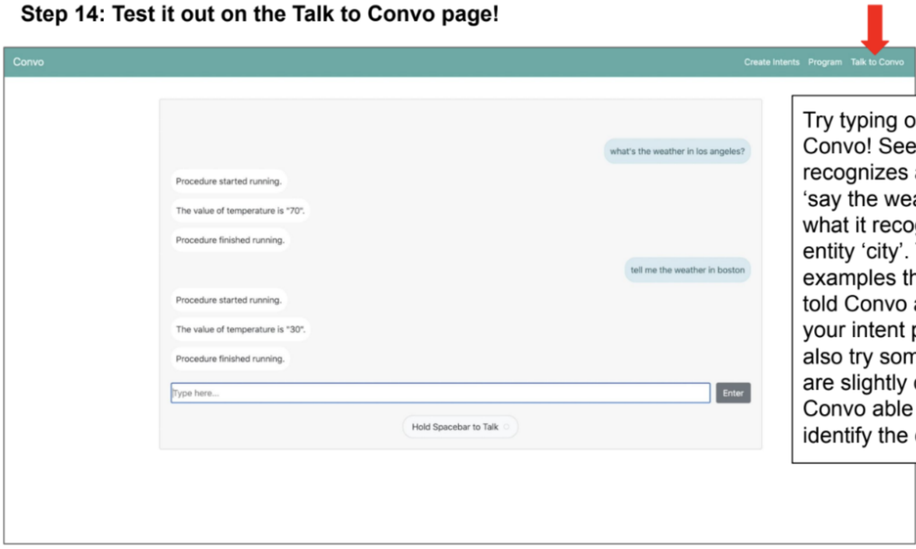
We add a couple of conditions. If Convo detects that a user is asking about boston, we set the temperature to a value of 30. If the city is los angeles, we set the temperature to a value of 70. You may choose to do different actions here.

Step 13 continued: Finish adding actions to the procedure.



We finish by saying the value of the variable temperature, which will be 30 if the city is boston, 70 if the city is los angeles, and 0 in all other cases. We indicate that we are done with adding actions by saying 'done'. Move to the Talk to Convo page to proceed.

Step 14: Test it out on the Talk to Convo page!



Try typing or talking to Convo! See what Convo recognizes as the intent ‘say the weather’, and what it recognizes as the entity ‘city’. Try saying examples that you directly told Convo about through your intent phrases, but also try some things that are slightly different. Is Convo able to always identify the correct city?

Bibliography

[1] Adam Agassi, Hadas Erel, Iddo Yehoshua Wald, and Oren Zuckerman. Scratch nodes ML: A playful system for children to create gesture recognition classifiers. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–6, Glasgow, Scotland, UK, 2019. ACM.

[2] Michal Al-Yagon and Mario Mikulincer. Socioemotional and academic adjustment among children with learning disorders: The mediational role of attachment-based factors. *The Journal of Special Education*, 38(2):111–123, 2004. doi: 10.1177/00224669040380020501. URL `https://doi.org/10.1177/00224669040380020501`.

[3] Mehdi Alaimi, Edith Law, Kevin Daniel Pantasdo, Pierre-Yves Oudeyer, and Hélène Sauzeon. *Pedagogical Agents for Fostering Question-Asking Skills in Children*, page 1–13. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450367080. URL `https://doi-org.libproxy.mit.edu/10.1145/3313831.3376776`.

[4] Joel Alejandro Mejia, Dustin Drake, and Amy Wilson-Lopez. Changes in latino/a adolescents' engineering self-efficacy and perceptions of engineering after addressing authentic engineering design challenges. In *2015 ASEE Annual Conference & Exposition*, Seattle, Washington, June 2015. ASEE Conferences. https://peer.asee.org/23678.

[5] Mark Alfano. Friendship and the structure of trust. In *From personality to virtue*, pages 186–206. Oxford University Press, 2016.

[6] Safinah Ali, Blakeley H Payne, Randi Williams, Hae Won Park, and Cynthia Breazeal. Constructionism, ethics, and creativity: Developing primary and middle school artificial intelligence education. In *International Workshop on Education in Artificial Intelligence K-12 (EDUAI'19)*, 2019.

[7] Merav Allouch, Amos Azaria, and Rina Azoulay. Conversational agents: Goals, technologies, vision and challenges. *Sensors*, 21(24), 2021. ISSN 1424-8220. doi: 10.3390/s21248448. URL `https://www.mdpi.com/1424-8220/21/24/8448`.

[8] Patrícia Alves, Pedro Campos, and Eugénio Oliveira. Modeling the trustworthiness of a supplier agent in a b2b relationship. In Luis M. Camarinha-Matos, Lai Xu, and Hamideh Afsarmanesh, editors, *Collaborative Networks in the Internet of Services*, pages 675–686, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-32775-9.

[9] Amazon. Skill blueprints. https://blueprints.amazon.com/, 2021. Accessed: 2021-09-14.

[10] Amazon. Alexa skills kit. https://developer.amazon.com/en-US/alexa/alexa-skills-kit, 2021. Accessed: 2021-10-26.

[11] Amazon. Get started with the guide. https://developer.amazon.com/en-US/docs/alexa/alexa-design/get-started.html, 2021. Accessed: 2021-10-15.

[12] Amazon. Alexa developer console. https://developer.amazon.com/alexa/console/ask, 2022. Accessed: 2022-05-23.

[13] Amazon. Build your Alexa connected device. https://developer.amazon.com/en-US/alexa/devices/connected-devices/development-resources, 2022. Accessed: 2022-05-18.

[14] Amazon. Tutorial: Build an engaging Alexa skill. https://developer.amazon.com/en-US/alexa/alexa-skills-kit/get-deeper/tutorials-code-samples/build-an-engaging-alexa-skill, 2022. Accessed: 2022-05-18.

[15] Apple. Sirikit. https://developer.apple.com/documentation/sirikit, 2022. Accessed: 2022-05-23.

[16] Ron Artstein and Massimo Poesio. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596, 12 2008. ISSN 0891-2017. doi: 10.1162/coli.07-034-R2. URL https://doi.org/10.1162/coli.07-034-R2.

[17] Petek Askar and David Davenport. An investigation of factors related to self-efficacy for java programming among engineering students. *Online Submission*, 8(1), 2009.

[18] Benett Axtell and Cosmin Munteanu. Tea, earl grey, hot: Designing speech interactions from the imagined ideal of Star Trek. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966.

[19] Dora M. Ballesteros, Yohanna Rodriguez-Ortega, Diego Renza, and Gonzalo Arce. Deep4SNet: Deep learning for fake speech classification. *Expert Systems with Applications*, 184:115465, 2021. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.115465. URL https://www.sciencedirect.com/science/article/pii/S0957417421008770.

[20] John Barryman. 18 Siri fails that will ruin your day. https://www.ranker.com/list/siri-fails-iphone-sucks/john-barryman, 2014. Accessed: 2021-09-01.

[21] Gabriele Barzilai and Lucia Rampino. Just a natural talk? the rise of intelligent personal assistants and the (hidden) legacy of ubiquitous computing. In Aaron Marcus and Elizabeth Rosenzweig, editors, *Design, User Experience, and Usability. Design for Contemporary Interactive Environments*, pages 18–39, Cham, 2020. Springer International Publishing. ISBN 978-3-030-49760-6.

[22] BayesianLearner. Apple: SiriKit looks dated before launch. https://seekingalpha.com/article/3991221-apple-sirikit-looks-dated-launch, 2016. Accessed: 2021-11-2.

[23] Izak Benbasat and Weiquan Wang. Trust in and adoption of online recommendation agents. *Journal of the association for information systems*, 6(3):4, 2005.

[24] Sondra H Birch and Gary W Ladd. The teacher-child relationship and children's early school adjustment. *Journal of school psychology*, 35(1):61–79, 1997.

[25] Gautam Biswas, Krittaya Leelawong, Daniel Schwartz, and Nancy Vye. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19:363–392, 03 2005. doi: 10.1080/08839510590910200.

[26] Gautam Biswas, James R Segedy, and Kritya Bunchongchit. From design to implementation to practice a learning by teaching system: Betty's brain. *International Journal of Artificial Intelligence in Education*, 26(1):350–364, 2016.

[27] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29:4349–4357, 2016.

[28] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[29] Marion Boulicault, Milo Phillips-Brown, Jacqueline M. Kory-Westlund, Stephanie Nguyen, and Cynthia Breazeal. *Authenticity and Co-design: On Responsibly Creating Relational Robots for Children*, chapter Connected Learning Summit, pages 1–61. MIT Press, jun 29 2021. https://wip.mitpress.mit.edu/pub/authenticity-and-co-design.

[30] Virginia Braun, Victoria Clarke, Nikki Hayfield, and Gareth Terry. Thematic analysis. In Pranee Liamputtong, editor, *Handbook of research methods in health social sciences*. Springer, Singapore, 2019.

[31] Karen Brennan and Mitchel Resnick. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, volume 1, page 25, 2012.

[32] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[33] Mark Campbell. Beyond conversational artificial intelligence. *Computer*, 53(12): 121–125, 2020. doi: 10.1109/MC.2020.3024401.

[34] Michelle Carney, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. Teachable machine: Approachable web-based tool for exploring machine learning classification. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–8, 2020.

[35] Carl Chapman and Kathryn T. Stolee. Exploring regular expression usage and context in python. In *Proceedings of the 25th International Symposium on Software Testing and Analysis*, ISSTA 2016, page 282–293, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450343909. doi: 10.1145/2931037.2931073. URL `https://doi.org/10.1145/2931037.2931073`.

[36] Thomas K. F. Chiu. Student engagement in K-12 online learning amid covid-19: A qualitative approach from a self-determination theory perspective. *Interactive Learning Environments*, 0(0):1–14, 2021. doi: 10.1080/10494820.2021.1926289. URL `https://doi.org/10.1080/10494820.2021.1926289`.

[37] Jin-Hee Cho, Kevin Chan, and Sibel Adali. A survey on trust modeling. *ACM Comput. Surv.*, 48(2), oct 2015. ISSN 0360-0300. doi: 10.1145/2815595. URL `https://doi.org/10.1145/2815595`.

[38] Leigh Clark, Nadia Pantidi, Orla Cooney, Philip Doyle, Diego Garaialde, Justin Edwards, Brendan Spillane, Emer Gilmartin, Christine Murad, Cosmin Munteanu, Vincent Wade, and Benjamin R. Cowan. *What Makes a Good Conversation? Challenges in Designing Truly Conversational Agents*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2019. ISBN 9781450359702. URL `https://doi.org/10.1145/3290605.3300705`.

[39] Jessie Cohen, Nicole Kerr, and Trina Dong. The peabody awards announce winners for digital and interactive storytelling. `https://peabodyawards.com/stories/the-peabody-awards-announce-winners-for-digital-and-interactive-storytelling/`, 2022. Accessed: 2022-06-20.

[40] AI4EU consortium. AI4EU. `https://www.ai4eu.eu/`, 2020. Accessed: 2021-09-11.

[41] Benjamin R. Cowan, Holly P. Branigan, Mateo Obregon, Enas Bugis, and Russell Beale. Voice anthropomorphism, interlocutor modelling and alignment effects on syntactic choices in human-computer dialogue. *International Journal of Human-Computer Studies*, 83:27–42, 2015. ISSN 1071-5819. doi: https://doi.org/10.1016/j.ijhcs.2015.05.008. URL `https://www.sciencedirect.com/science/article/pii/S1071581915001020`.

[42] Benjamin R Cowan, Holly P Branigan, Habiba Begum, Lucy McKenna, and Eva Szekely. They know as much as we do: Knowledge estimation and partner modelling of artificial partners. In *The Annual Meeting of the Cognitive Science Society (COGSCI)*, 2017.

[43] Benjamin R. Cowan, Philip Doyle, Justin Edwards, Diego Garaialde, Ali Hayes-Brady, Holly P. Branigan, João Cabral, and Leigh Clark. What's in an accent? the impact of accented synthetic speech on lexical choice in human-machine dialogue. In *Proceedings of the 1st International Conference on Conversational User Interfaces*, CUI '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450371872. doi: 10.1145/3342775.3342786. URL `https://doi.org/10.1145/3342775.3342786`.

[44] Stephanie Craft, Seth Ashley, and Adam Maksl. News media literacy and conspiracy theory endorsement. *Communication and the Public*, 2(4):388–401, 2017. doi: 10.1177/2057047317725539. URL `https://doi.org/10.1177/2057047317725539`.

[45] Andrew Cunningham. The abilities and limitations of SiriKit, Siri's olive branch to other apps. https://arstechnica.com/gadgets/2016/06/the-abilities-and-limitations-of-sirikit-siris-olive-branch-to-other-apps/, 2016. Accessed: 2021-11-2.

[46] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[47] Daniella DiPaola. How does my robot know who i am?: Understanding the impact of education on child-robot relationships. Master's thesis, Massachusetts Institute of Technology, 2021.

[48] Daniella DiPaola, Blakeley H Payne, and Cynthia Breazeal. Decoding design agendas: An ethical design activity for middle school students. In *Proceedings of the Interaction Design and Children Conference*, pages 1–10, 2020.

[49] Philip R Doyle, Leigh Clark, and Benjamin R. Cowan. What do we see in them? identifying dimensions of partner models for speech interfaces using a psycholexical approach. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445206. URL https://doi.org/10.1145/3411764.3445206.

[50] Stefania Druga. Growing up with AI: Cognimates: from coding to teaching machines. Master's thesis, Massachusetts Institute of Technology, 2018.

[51] Stefania Druga. Cognimates: My favorite things. http://cognimates.me/projects/myFavoriteThings, 2019. Accessed: 2021-09-14.

[52] Stefania Druga, Randi Williams, Cynthia Breazeal, and Mitchel Resnick. "hey google is it ok if i eat you?": Initial explorations in child-agent interaction. In *Proceedings of the 2017 Conference on Interaction Design and Children*, IDC '17, page 595–600, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349215. doi: 10.1145/3078072.3084330. URL https://doi.org/10.1145/3078072.3084330.

[53] Stefania Druga, Sarah T. Vu, Eesh Likhith, and Tammy Qiu. Inclusive AI literacy for kids around the world. In *Proceedings of FabLearn 2019*, FL2019, page 104–111, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362443. doi: 10.1145/3311890.3311904. URL https://doi.org/10.1145/3311890.3311904.

[54] Hui Fang, Jie Zhang, Murat Sensoy, and Nadia Magnenat Thalmann. A reputation mechanism for virtual reality - five-sense oriented feedback provision and subjectivity alignment. In *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 312–319, 2011. doi: 10.1109/TrustCom.2011.42.

[55] Hui Fang, Jie Zhang, Murat Şensoy, and Nadia Magnenat-Thalmann. Reputation mechanism for e-commerce in virtual reality environments. *Electronic Commerce Research and Applications*, 13(6):409–422, 2014. ISSN 1567-4223. doi:

https://doi.org/10.1016/j.elerap.2014.08.002. URL https://www.sciencedirect.com/science/article/pii/S1567422314000490.

[56] Carlos Flavián, Miguel Guinalíu, and Raquel Gurrea. The role played by perceived usability, satisfaction and consumer trust on website loyalty. *Information & Management*, 43(1):1–14, 2006. ISSN 0378-7206. doi: https://doi.org/10.1016/j.im.2005.01.002. URL https://www.sciencedirect.com/science/article/pii/S0378720605000169.

[57] Lokesh Fulfagar, Anupriya Gupta, Arpit Mathur, and Abhishek Shrivastava. Development and evaluation of usability heuristics for voice user interfaces. In Amaresh Chakrabarti, Ravi Poovaiah, Prasad Bokil, and Vivek Kant, editors, *Design for Tomorrow—Volume 1*, pages 375–385, Singapore, 2021. Springer Singapore. ISBN 978-981-16-0041-8.

[58] Radhika Garg, Hua Cui, Spencer Seligson, Bo Zhang, Martin Porcheron, Leigh Clark, Benjamin R. Cowan, and Erin Beneteau. The last decade of hci research on children and voice-based conversational agents. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3502016. URL https://doi.org/10.1145/3491102.3502016.

[59] Susanne Gaube, Harini Suresh, Martina Raue, Alexander Merritt, Seth J. Berkowitz, Eva Lermer, Joseph F. Coughlin, John V. Guttag, Errol Colak, and Marzyeh Ghassemi. Do as AI say: susceptibility in deployment of clinical decision-aids. *npj Digital Medicine*, 4(1):31, Feb 2021. ISSN 2398-6352. doi: 10.1038/s41746-021-00385-9. URL https://doi.org/10.1038/s41746-021-00385-9.

[60] Google. Introduction to the speech synthesis api, 2014. https://developers.google.com/web/updates/2014/01/Web-apps-that-talk-Introduction-to-the-Speech-Synthesis-API, Last accessed on 2020-02-24.

[61] Google. Google cloud speech-to-text, 2020. https://cloud.google.com/speech-to-text, Last accessed on 2020-02-24.

[62] Google. Dialogflow cx. https://dialogflow.cloud.google.com/cx/, 2021. Accessed: 2021-10-15.

[63] Google. Dialogflow essentials. https://dialogflow.cloud.google.com/, 2021. Accessed: 2021-10-15.

[64] Google. Learn about conversation. https://developers.google.com/assistant/conversation-design/learn-about-conversation, 2021. Accessed: 2021-10-15.

[65] Simon Gray, Caroline St. Clair, Richard James, and Jerry Mead. Suggestions for graduated exposure to programming concepts using fading worked examples. In *Proceedings of the third international workshop on Computing education research*, pages 99–110, 2007.

[66] Thomas RG Green. Cognitive dimensions of notations. *People and computers V*, pages 443–460, 1989.

[67] Anne Gregory and Michael B. Ripski. Adolescent trust in teachers: Implications for behavior in the high school classroom. *School Psychology Review*, 37(3):337–353, 2008. doi: 10.1080/02796015.2008.12087881. URL https://doi.org/10.1080/02796015.2008.12087881.

[68] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. In *21st Annual Conference of the International Speech Communication Association (Interspeech 2020)*, 2020.

[69] Karen Haoarchive. The future of AI research is in Africa. https://www.technologyreview.com/2019/06/21/134820/ai-africa-machine-learning-ibm-google/, 2019. Accessed: 2021-09-13.

[70] Steve Harlow, Rhoda Cummings, and Suzanne M. Aberasturi. Karl popper and jean piaget: A rationale for constructivism. *The Educational Forum*, 71(1):41–48, 2007. doi: 10.1080/00131720608984566. URL https://doi.org/10.1080/00131720608984566.

[71] Yugo Hayashi. Gaze awareness and metacognitive suggestions by a pedagogical conversational agent: an experimental investigation on interventions to support collaborative learning process and performance. *International Journal of Computer-Supported Collaborative Learning*, 15(4):469–498, Dec 2020. ISSN 1556-1615. doi: 10.1007/s11412-020-09333-3. URL https://doi.org/10.1007/s11412-020-09333-3.

[72] Clint Andrew Heinze, Janet Haase, and Helen Higgins. An action research report from a multi-year approach to teaching artificial intelligence at the K-6 level. In *First AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI)*, 2010.

[73] Wesley Hilliard. Apple deprecates some third-party Siri apis starting in ios 15, macos monterey. https://appleinsider.com/articles/21/07/28/apple-deprecates-some-third-party-siri-apis-starting-in-ios-15-macos-monterey, 2021. Accessed: 2021-11-2.

[74] Pamela J Hinds, Kathleen M Carley, David Krackhardt, and Doug Wholey. Choosing work group members: Balancing similarity, competence, and familiarity. *Organizational Behavior and Human Decision Processes*, 81(2):226–251, 2000. ISSN 0749-5978. doi: https://doi.org/10.1006/obhd.1999.2875. URL https://www.sciencedirect.com/science/article/pii/S0749597899928753.

[75] Joshua WK Ho and Matthew Scadding. Classroom activities for teaching artificial intelligence to primary school students. *CoolThink@ JC*, page 157, 2019.

[76] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

[77] Matthew Huggins, Sharifa Alghowinem, Sooyeon Jeong, Pedro Colon-Hernandez, Cynthia Breazeal, and Hae Won Park. Practical guidelines for intent recognition: Bert with minimal training data evaluated in real-world hri application. In *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '21, page 341–350, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382892. doi: 10.1145/3434073.3444671. URL `https://doi.org/10.1145/3434073.3444671`.

[78] Matthew Huggins, Anastasia K Ostrowski, Andrew Rapo, Eric Woudenberg, Cynthia Breazeal, and Hae Won Park. The interaction flow editor: A new human-robot interaction rapid prototyping interface. *arXiv preprint arXiv:2108.13838*, 2021.

[79] Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. Direct manipulation interfaces. *Human–Computer Interaction*, 1(4):311–338, 1985. doi: 10.1207/s15327051hci0104_2. URL `https://doi.org/10.1207/s15327051hci0104_2`.

[80] Sheila Jasanoff and Sang-Hyun Kim, editors. *Dreamscapes of Modernity: Sociotechnical Imaginaries and the Fabrication of Power*. University of Chicago Press, 2015. doi: doi:10.7208/9780226276663.

[81] W. Lewis Johnson and James C. Lester. Pedagogical agents: Back to the future. *AI Magazine*, 39(2):33–44, Jul. 2018. doi: 10.1609/aimag.v39i2.2793. URL `https://ojs.aaai.org/index.php/aimagazine/article/view/2793`.

[82] Sonam Joshi. Teenage girl coders from mumbai slum are building apps to solve local problems. `https://mashable.com/archive/mumbai-dharavi-girls-coding-apps`, 2016. Accessed: 2022-03-04.

[83] K Megasari Kahn, Rani Megasari, Erna Piantari, and Enjun Junaeti. AI programming by children using Snap! block programming in a developing country. In *EC-TEL 2018*. Springer, 2018.

[84] Maria Kallia and Sue Sentance. Are boys more confident than girls? the role of calibration and students' self-efficacy in programming tasks and computer science. In *Proceedings of the 13th Workshop in Primary and Secondary Computing Education*, WiPSCE '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450365888. doi: 10.1145/3265757.3265773. URL `https://doi.org/10.1145/3265757.3265773`.

[85] Yelim Kim, Mohi Reza, Joanna McGrenere, and Dongwook Yoon. Designers characterize naturalness in voice user interfaces: Their goals, practices, and challenges. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445579. URL `https://doi.org/10.1145/3411764.3445579`.

[86] Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R. Rickford, Dan Jurafsky, and Sharad Goel. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences*, 117(14):7684–7689, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1915768117. URL `https://www.pnas.org/content/117/14/7684`.

[87] Jacqueline M. Kory-Westlund and Cynthia Breazeal. Exploring the effects of a social robot's speech entrainment and backstory on young children's emotion, rapport, relationship, and learning. *Frontiers in Robotics and AI*, 6, 2019. ISSN 2296-9144. doi: 10.3389/frobt.2019.00054. URL https://www.frontiersin.org/articles/10.3389/frobt.2019.00054.

[88] Jacqueline M. Kory-Westlund and Cynthia Breazeal. A long-term study of young children's rapport, social emulation, and language learning with a peer-like robot playmate in preschool. *Frontiers in Robotics and AI*, 6, 2019. ISSN 2296-9144. doi: 10.3389/frobt.2019.00081. URL https://www.frontiersin.org/articles/10.3389/frobt.2019.00081.

[89] Jacqueline Marie Kory-Westlund. *Relational AI: Creating long-term interpersonal interaction, rapport, and relationships with social robots*. PhD thesis, Massachusetts Institute of Technology, 2019.

[90] Kushal Lakhotia, Evgeny Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Adelrahman Mohamed, et al. Generative spoken language modeling from raw audio. *arXiv preprint arXiv:2102.01192*, 2021.

[91] Dale Lane. Machine learning for kids. https://machinelearningforkids.co.uk/, 2020. Accessed: 2020-09-05.

[92] Raina Langevin, Ross J Lordon, Thi Avrahami, Benjamin R. Cowan, Tad Hirsch, and Gary Hsieh. Heuristic evaluation of conversational agents. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445312. URL https://doi.org/10.1145/3411764.3445312.

[93] N. Lao, I. Lee, and H. Abelson. A deep learning practicum: Concepts and practices for teaching actionable machine learning at the tertiary education level. In *ICERI2019 Proceedings*, 12th annual International Conference of Education, Research and Innovation, pages 405–415. IATED, 11-13 November, 2019 2019. ISBN 978-84-09-14755-7. doi: 10.21125/iceri.2019.0137. URL https://dx.doi.org/10.21125/iceri.2019.0137.

[94] Natalie Lao. CloudDB: Components for exploring shared data with MIT App Inventor. In *2017 IEEE Blocks and Beyond Workshop (B&B)*, pages 109–110. IEEE, 2017.

[95] Natalie Lao. *Reorienting Machine Learning Education Towards Tinkerers and ML-Engaged Citizens*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 9 2020.

[96] Edith Law, Parastoo Baghaei Ravari, Nalin Chhibber, Dana Kulic, Stephanie Lin, Kevin D. Pantasdo, Jessy Ceha, Sangho Suh, and Nicole Dillen. Curiosity notebook: A platform for learning by teaching conversational agents. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI EA '20, page 1–9, New York, NY, USA, 2020. Association for Computing Machinery.

ISBN 9781450368193. doi: 10.1145/3334480.3382783. URL https://doi.org/10.1145/3334480.3382783.

[97] Eunbae Lee and Michael J. Hannafin. A design framework for enhancing engagement in student-centered learning: own it, learn it, and share it. *Educational Technology Research and Development*, 64(4):707–734, Aug 2016. ISSN 1556-6501. doi: 10.1007/s11423-015-9422-5. URL https://doi.org/10.1007/s11423-015-9422-5.

[98] Toby Jia-Jun Li, Igor Labutov, Brad A. Myers, Amos Azaria, Alexander I. Rudnicky, and Tom M. Mitchell. *Teaching Agents When They Fail: End User Development in Goal-Oriented Conversational Agents*, pages 119–137. Springer International Publishing, Cham, 2018. ISBN 978-3-319-95579-7. doi: 10.1007/978-3-319-95579-7_6. URL https://doi.org/10.1007/978-3-319-95579-7_6.

[99] Phoebe Lin, Jessica Van Brummelen, Galit Lukin, Randi Williams, and Cynthia Breazeal. Zhorai: Designing a conversational agent for children to explore machine learning concepts. In *AAAI*, pages 13381–13388, 2020.

[100] Duri Long and Brian Magerko. What is AI literacy? competencies and design considerations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–16, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450367080. doi: 10.1145/3313831.3376727. URL https://doi.org/10.1145/3313831.3376727.

[101] Gustavo López, Luis Quesada, and Luis A. Guerrero. Alexa vs. Siri vs. Cortana vs. Google Assistant: A comparison of speech-based natural user interfaces. In Isabel L. Nunes, editor, *Advances in Human Factors and Systems Interaction*, pages 241–250, Cham, 2018. Springer International Publishing. ISBN 978-3-319-60366-7.

[102] Silvia B. Lovato, Anne Marie Piper, and Ellen A. Wartella. Hey google, do unicorns exist? conversational agents as a path to answers to children's questions. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, IDC '19, page 301–313, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366908. doi: 10.1145/3311927.3323150. URL https://doi.org/10.1145/3311927.3323150.

[103] Siu Man Lui and Wendy Hui. Effects of smiling and gender on trust toward a recommendation agent. In *2010 International Conference on Cyberworlds*, pages 398–405, 2010. doi: 10.1109/CW.2010.26.

[104] Martin Maguire. Development of a heuristic evaluation tool for voice user interfaces. In Aaron Marcus and Wentao Wang, editors, *Design, User Experience, and Usability. Practice and Case Studies*, pages 212–225, Cham, 2019. Springer International Publishing. ISBN 978-3-030-23535-2.

[105] Radu Mariescu-Istodor and Ilkka Jormanainen. Machine learning for high school students. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, Koli Calling '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450377157. doi: 10.1145/3364510.3364520. URL https://doi.org/10.1145/3364510.3364520.

[106] Chris Mayo. promptly-dotnet. `https://github.com/cmayomsft/promptly-dotnet`, 2018. Accessed: 2021-11-2.

[107] D. Harrison McKnight and Norman L. Chervany. What trust means in e-commerce customer relationships: An interdisciplinary conceptual typology. *International Journal of Electronic Commerce*, 6(2):35–59, 2001. doi: 10.1080/10864415.2001.11044235. URL `https://doi.org/10.1080/10864415.2001.11044235`.

[108] Shari J. Metcalf, Joseph M. Reilly, Soobin Jeon, Annie Wang, Allyson Pyers, Karen Brennan, and Chris Dede. Assessing computational thinking through the lenses of functionality and computational fluency. *Computer Science Education*, 31(2):199–223, 2021. doi: 10.1080/08993408.2020.1866932. URL `https://doi.org/10.1080/08993408.2020.1866932`.

[109] Joseph E. Michaelis and Bilge Mutlu. Supporting interest in science learning with a social robot. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, IDC '19, page 71–82, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366908. doi: 10.1145/3311927.3323154. URL `https://doi.org/10.1145/3311927.3323154`.

[110] Miro. Miro: The leading visual collaboration platform. `https://miro.com/about/`, 2022. Accessed: 2022-06-29.

[111] MIT. Responsible AI for social empowerment and education (RAISE). `https://raise.mit.edu/index.html`, 2021. Accessed: 2021-09-11.

[112] MIT App Inventor. Artificial intelligence with mit app inventor. `https://appinventor.mit.edu/explore/ai-with-mit-app-inventor`, 2020. Accessed: 2020-09-10.

[113] Roxanne M. Mitchell, Lisa Kensler, and Megan Tschannen-Moran. Student trust in teachers and student perceptions of safety: positive predictors of student identification with school. *International Journal of Leadership in Education*, 21(2):135–154, 2018. doi: 10.1080/13603124.2016.1157211. URL `https://doi.org/10.1080/13603124.2016.1157211`.

[114] Masahiro Mori, Karl F. MacDorman, and Norri Kageki. The uncanny valley [from the field]. *IEEE Robotics Automation Magazine*, 19(2):98–100, 2012. doi: 10.1109/MRA.2012.2192811.

[115] Christine Murad and Cosmin Munteanu. Alexa, how do i build a VUI curriculum? In *Proceedings of the 2nd Conference on Conversational User Interfaces*, CUI '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450375443. doi: 10.1145/3405755.3406137. URL `https://doi.org/10.1145/3405755.3406137`.

[116] Christine Murad and Cosmin Munteanu. Designing voice interfaces: Back to the (curriculum) basics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, page 1–12, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450367080. doi: 10.1145/3313831.3376522. URL `https://doi.org/10.1145/3313831.3376522`.

[117] Christine Murad, Cosmin Munteanu, Leigh Clark, and Benjamin R. Cowan. Design guidelines for hands-free speech interaction. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, MobileHCI '18, page 269–276, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450359412. doi: 10.1145/3236112.3236149. URL `https://doi-org.libproxy.mit.edu/10.1145/3236112.3236149`.

[118] Christine Murad, Cosmin Munteanu, Benjamin R. Cowan, and Leigh Clark. Revolution or evolution? speech interaction and hci design guidelines. *IEEE Pervasive Computing*, 18(2):33–45, 2019. doi: 10.1109/MPRV.2019.2906991.

[119] Christine Murad, Cosmin Munteanu, Benjamin R. Cowan, and Leigh Clark. Finding a new voice: Transitioning designers from gui to vui design. In *CUI 2021 - 3rd Conference on Conversational User Interfaces*, CUI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450389983. doi: 10.1145/3469595.3469617. URL `https://doi.org/10.1145/3469595.3469617`.

[120] Clifford Ivar Nass and Scott Brave. *Wired for speech: How voice activates and advances the human-computer relationship*. MIT press Cambridge, 2005.

[121] NGSS. Next generation science standards: 5-LS2 ecosystems, 2016. `https://www.nextgenscience.org/`, Last accessed on 2019-11-15.

[122] J. Nielsen and R. Molich. Teaching user interface design based on usability engineering. *SIGCHI Bull.*, 21(1):45–48, aug 1989. ISSN 0736-6906. doi: 10.1145/67880.67885. URL `https://doi.org/10.1145/67880.67885`.

[123] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, page 152–158, New York, NY, USA, 1994. Association for Computing Machinery. ISBN 0897916506. doi: 10.1145/191666.191729. URL `https://doi.org/10.1145/191666.191729`.

[124] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, page 152–158, New York, NY, USA, 1994. Association for Computing Machinery. ISBN 0897916506. doi: 10.1145/191666.191729. URL `https://doi.org/10.1145/191666.191729`.

[125] Jakob Nielsen and Rolf Molich. Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, page 249–256, New York, NY, USA, 1990. Association for Computing Machinery. ISBN 0201509326. doi: 10.1145/97243.97281. URL `https://doi.org/10.1145/97243.97281`.

[126] Don Norman. *The design of everyday things: Revised and expanded edition*. Basic books, 2013.

[127] Donald A Norman. Natural user interfaces are not natural. *interactions*, 17(3): 6–10, 2010.

[128] Narges Norouzi, Snigdha Chaturvedi, and Matthew Rutledge. Lessons learned from teaching machine learning and natural language processing to high school students. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(09):13397–13403, Apr. 2020. doi: 10.1609/aaai.v34i09.7063. URL `https://ojs.aaai.org/index.php/AAAI/article/view/7063`.

[129] Caroline Nowacki, Anna Gordeeva, and Anne-Hélène Lizé. Improving the usability of voice user interfaces: A new set of ergonomic criteria. In Aaron Marcus and Elizabeth Rosenzweig, editors, *Design, User Experience, and Usability. Design for Contemporary Interactive Environments*, pages 117–133, Cham, 2020. Springer International Publishing. ISBN 978-3-030-49760-6.

[130] Department of International Cooperation Ministry of Science and Technology. Next generation artificial intelligence development plan. `http://fi.china-embassy.org/eng/kxjs/P020171025789108009001.pdf`, 2017. Accessed: 2021-09-11.

[131] Simone Opel, Michael Schlichtig, and Carsten Schulte. Developing teaching materials on artificial intelligence by using a simulation game (work in progress). In *Proceedings of the 14th Workshop in Primary and Secondary Computing Education*, pages 1–2, 2019.

[132] Carol Xiaojuan Ou and Choon Ling Sia. Consumer trust and distrust: An issue of website design. *International Journal of Human-Computer Studies*, 68(12):913–934, 2010. ISSN 1071-5819. doi: https://doi.org/10.1016/j.ijhcs.2010.08.003. URL `https://www.sciencedirect.com/science/article/pii/S107158191000100X`.

[133] Laxmi Pandey, Khalad Hasan, and Ahmed Sabbir Arif. Acceptability of speech and silent speech input methods in private and public. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445430. URL `https://doi.org/10.1145/3411764.3445430`.

[134] Hannah (Nicole) Pang. Computational action in action: Process and tools that empower students to make a real-world impact using technology. Master's thesis, Massachusetts Institute of Technology, 2022.

[135] Seymour Papert and Idit Harel. Situating constructionism. *Constructionism*, 36 (2):1–11, 1991.

[136] Seymour A Papert. *Mindstorms: Children, computers, and powerful ideas*. Basic books, 2020.

[137] Evan Patton, Druga Stefania, and Jessica Van Brummelen. Panel: Block abstractions for AI. In *BLOCKS+ at SPLASH*, 2018.

[138] B.H. Payne. Can my algorithm be my opinion?: An AI + ethics curriculum for middle school students. Master's thesis, Massachusetts Institute of Technology, Media Lab, Cambridge, MA, USA., 2020.

[139] Supavich (Fone) Pengnate and Rathindra Sarathy. An experimental investigation of the influence of website emotional design features on trust in unfamiliar online vendors. *Computers in Human Behavior*, 67:49–60, 2017. ISSN 0747-5632. doi:

https://doi.org/10.1016/j.chb.2016.10.018. URL https://www.sciencedirect.com/science/article/pii/S0747563216307191.

[140] Gustav Bøg Petersen, Aske Mottelson, and Guido Makransky. Pedagogical agents in educational vr: An in the wild study. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. URL https://doi.org/10.1145/3411764.3445760.

[141] The AI Education Project. We're preparing all students to thrive in the age of artificial intelligence. https://aiedu.org/about-us, 2021. Accessed: 2021-09-13.

[142] Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, Eric King, Kate Bland, Amanda Wartick, Yi Pan, Han Song, Sk Jayadevan, Gene Hwang, and Art Pettigrue. Conversational AI: The science behind the Alexa prize. *arXiv*, 2018. doi: 10.48550/ARXIV.1801.03604. URL https://arxiv.org/abs/1801.03604.

[143] Partha Pratim Ray. A survey on visual programming languages in internet of things. *Scientific Programming*, 2017, 2017.

[144] ReadyAI. ReadyAI. https://www.readyai.org/, 2021. Accessed: 2021-09-13.

[145] Reddit. Alexa misunderstandings... https://www.reddit.com/r/amazonecho/comments/41159u/alexa_misunderstandings/, 2015. Accessed: 2021-09-01.

[146] Zobia Rehman and Stefania Kifor. Teaching natural language processing (nlp) using ontology based education design. *Balkan Region Conference on Engineering and Business Education*, 1(1):206–214, 2015. doi: doi:10.1515/cplbu-2015-0024. URL https://doi.org/10.1515/cplbu-2015-0024.

[147] RobotLab. Zumi classroom pack: Robotics, coding & AI. https://www.robotlab.com/store/zumi-classroom-pack, 2021. Accessed: 2021-09-13.

[148] Juan David Rodríguez-García, Jesús Moreno-León, Marcos Román-González, and Gregorio Robles. Evaluation of an online intervention to teach artificial intelligence with LearningML to 10-16-year-old students. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, SIGCSE '21, page 177–183, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380621. doi: 10.1145/3408877.3432393. URL https://doi.org/10.1145/3408877.3432393.

[149] Elayne Ruane, Abeba Birhane, and Anthony Ventresque. Conversational AI: Social and ethical considerations. In *AICS*, pages 104–115, 2019.

[150] Elayne Ruane, Abeba Birhane, and Anthony Ventresque. Conversational AI: Social and ethical considerations. In *AICS*, pages 104–115, 2019.

[151] Christine Rzepka, Benedikt Berger, and Thomas Hess. Voice assistant vs. chatbot – examining the fit between conversational agents' interaction modalities and information search tasks. *Information Systems Frontiers*, Dec 2021. ISSN 1572-9419. doi: 10.1007/s10796-021-10226-5. URL https://doi.org/10.1007/s10796-021-10226-5.

[152] Alpay Sabuncuoglu. Designing one year curriculum to teach artificial intelligence for middle school. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*, pages 96–102, 2020.

[153] Bawornsak Sakulkueakulsuk, Siyada Witoon, Potiwat Ngarmkajornwiwat, Pornpen Pataranutaporn, Werasak Surareungchai, Pat Pataranutaporn, and Pakpoom Subsoontorn. Kids making AI: Integrating machine learning, gamification, and social context in STEM education. In *2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pages 1005–1010. IEEE, 2018.

[154] Alexander Scheidt and Tim Pulver. Any-cubes: A children's toy for learning AI: Enhanced play with deep learning and mqtt. In *Proceedings of Mensch Und Computer 2019*, MuC'19, page 893–895. Association for Computing Machinery, New York, NY, USA, 2019. ISBN 9781450371988. doi: 10.1145/3340764.3345375. URL `https://doi.org/10.1145/3340764.3345375`.

[155] Sofia Schöbel, Andreas Janson, and Abhay Mishra. A configurational view on avatar design–the role of emotional attachment, satisfaction, and cognitive load in digital learning. In *Fortieth International Conference on Information Systems, Munich*, 2019.

[156] Oscar Schwartz. Why people demanded privacy to confide in the world's first chatbot. `https://spectrum.ieee.org/why-people-demanded-privacy-to-confide-in-the-worlds-first-chatbot`, 2019. Accessed: 2022-06-20.

[157] Scratch. Creating a chat bot. `https://en.scratch-wiki.info/wiki/Creating_a_Chat_Bot`, 2021. Accessed: 2021-09-14.

[158] Katie Seaborn, Norihisa P. Miyake, Peter Pennefather, and Mihoko Otake-Matsuura. Voice in human–agent interaction: A survey. *ACM Comput. Surv.*, 54(4), May 2021. ISSN 0360-0300. doi: 10.1145/3386867. URL `https://doi.org/10.1145/3386867`.

[159] Haeseung Seo, Aiping Xiong, and Dongwon Lee. Trust it or not: Effects of machine-learning warnings in helping individuals mitigate misinformation. In *Proceedings of the 10th ACM Conference on Web Science*, WebSci '19, page 265–274, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362023. doi: 10.1145/3292522.3326012. URL `https://doi.org/10.1145/3292522.3326012`.

[160] William Seymour and Max Van Kleek. Exploring interactions between trust, anthropomorphism, and relationship development in voice assistants. *Proc. ACM Hum.-Comput. Interact.*, 5(CSCW2), oct 2021. doi: 10.1145/3479515. URL `https://doi.org/10.1145/3479515`.

[161] Ben. Shneiderman. *Designing the user interface : strategies for effective human-computer-interaction.* Addison Wesley Longman, Reading, Mass, 3rd ed. edition, 1998. ISBN 0201694972.

[162] Ben Shneiderman and Catherine Plaisant. *Designing the user interface: Strategies for effective human-computer interaction.* Pearson Education India, 2010.

[163] Sam Smith. Digital voice assistants in use to triple to 8 billion by 2023, driven by smart home devices. Scientific analysis or review, Juniper Research, Hampshire, UK, 2018. URL https://www.juniperresearch.com/press/digital-voice-assistants-in-use-to-8-million-2023.

[164] Shirish C. Srivastava and Shalini Chandra. Social presence in virtual world collaboration: An uncertainty reduction perspective using a mixed methods approach. *MIS Q.*, 42(3):779–804, sep 2018. ISSN 0276-7783. doi: 10.25300/MISQ/2018/11914. URL https://doi.org/10.25300/MISQ/2018/11914.

[165] Statista. Voice commerce in the united states, 2021. URL https://www.statista.com/study/60607/voice-commerce-in-the-united-states/.

[166] Caroline L. van Straten, Jochen Peter, Rinaldo Kühne, and Alex Barco. Transparency about a robot's lack of human psychological capacities: Effects on child-robot perception and relationship formation. *J. Hum.-Robot Interact.*, 9(2), jan 2020. doi: 10.1145/3365668. URL https://doi.org/10.1145/3365668.

[167] Christian Sturm, Alice Oh, Sebastian Linxen, Jose Abdelnour Nocera, Susan Dray, and Katharina Reinecke. How weird is hci? extending hci principles to other countries and cultures. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, page 2425–2428, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450331463. doi: 10.1145/2702613.2702656. URL https://doi.org/10.1145/2702613.2702656.

[168] Kyoko Sugisaki and Andreas Bleiker. Usability guidelines and evaluation criteria for conversational user interfaces: A heuristic and linguistic approach. In *Proceedings of the Conference on Mensch Und Computer*, MuC '20, page 309–319, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450375405. doi: 10.1145/3404983.3405505. URL https://doi.org/10.1145/3404983.3405505.

[169] Danny Tang. Empowering novices to understand and use machine learning with personalized image classification models, intuitive analysis tools, and mit app inventor. Master's thesis, Massachusetts Institute of Technology, 77 Massachusetts Ave, Cambridge, MA, USA, 2019.

[170] Danny Tang, Yuria Utsumi, and Natalie Lao. PIC: A personal image classification webtool for high school students. In *Proceedings of the 2019 IJCAI EduAI Workshop*. IJCAI, 2019.

[171] Dan Tappan. A pedagogical framework for modeling and simulating intelligent agents and control systems. *Technical Report WS-08-02*, 2008.

[172] Rachael Tatman. Gender and dialect bias in YouTube's automatic captions. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 53–59, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1606. URL https://aclanthology.org/W17-1606.

[173] Google Cloud Tech. Introduction videos. https://cloud.google.com/dialogflow/cx/docs/video, 2021. Accessed: 2022-04-22.

[174] Google Cloud Tech. General agent design best practices. https://cloud.google.com/dialogflow/cx/docs/concept/agent-design, 2021. Accessed: 2022-04-30.

[175] Google Cloud Tech. Voice agent design best practices. https://cloud.google.com/dialogflow/cx/docs/concept/voice-agent-design, 2021. Accessed: 2022-04-30.

[176] Mike Tissenbaum, Josh Sheldon, and Hal Abelson. From computational thinking to computational action. *Commun. ACM*, 62(3):34–36, feb 2019. ISSN 0001-0782. doi: 10.1145/3265747. URL https://doi.org/10.1145/3265747.

[177] Mike Tissenbaum, David Weintrop, Nathan Holbert, and Tamara Clegg. The case for alternative endpoints in computing education. *British Journal of Educational Technology*, 52(3):1164–1177, 2021. doi: https://doi.org/10.1111/bjet.13072. URL https://bera-journals.onlinelibrary.wiley.com/doi/abs/10.1111/bjet.13072.

[178] David Touretzky, Christina Gardner-McCune, Fred Martin, and Deborah Seehorn. Envisioning AI for K-12: What should every child know about AI? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9795–9799, 2019.

[179] David S Touretzky and Christina Gardner-McCune. Calypso for cozmo: Robotic AI for everyone. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, pages 1110–1110, Baltimore, MD, USA, 2018. ACM.

[180] Vicki Trowler. Student engagement literature review. *The higher education academy*, 11(1):1–15, 2010.

[181] Sherry Turkle. *The Empathy Diaries: A Memoir*. Penguin, 2021.

[182] Charisi V, Chaudron S, Di Gioia R, Vuorikari R, Escobar Planas M, Sanchez Martin JI, and Gomez Gutierrez E. Artificial intelligence and the rights of the child: Towards an integrated agenda for research and policy. Scientific analysis or review KJ-NA-31048-EN-N (online), Joint Research Centre, Luxembourg (Luxembourg), 2022. URL https://publications.jrc.ec.europa.eu/repository/handle/JRC127564.

[183] Giedrė Vaičiulaitytė. 25 funny tweets about Amazon Alexa that prove there's nothing artificial about her intelligence. https://www.boredpanda.com/funny-alexa-tweets/?utm_source=google&utm_medium=organic&utm_campaign=organic, 2018. Accessed: 2021-09-01.

[184] Jessica Van Brummelen. Conversational agents to democratize artificial intelligence. In *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 239–240. IEEE, 2019.

[185] Jessica Van Brummelen. Conversational artificial intelligence development tools for K-12 education. In *2019 AAAI Fall Symposium*, 2019.

[186] Jessica Van Brummelen. Tools to create and democratize conversational artificial intelligence. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2019.

[187] Jessica Van Brummelen. Appendix for Convo: What does conversational programming mean?, 2020. `https://gist.github.com/jessvb/30271c1b7885a517296d1d30c566ea3f`, Last accessed on 2020-05-20.

[188] Jessica Van Brummelen and Phoebe Lin. Engaging teachers to co-design integrated AI curriculum for K-12 classrooms. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445377. URL `https://doi.org/10.1145/3411764.3445377`.

[189] Jessica Van Brummelen, Judy Hanwen Shen, and Evan W Patton. The popstar, the poet, and the Grinch: Relating artificial intelligence to the computational thinking framework with block-based coding. In *Proceedings of International Conference on Computational Thinking Education*, volume 3, pages 160–161, 2019.

[190] Jessica Van Brummelen, Kevin Weng, Phoebe Lin, and Catherine Yeo. Convo: What does conversational programming need? In *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 1–5. IEEE, 2020.

[191] Jessica Van Brummelen, Tommy Heng, and Viktoriya Tabunshchyk. Teaching tech to talk: K-12 conversational artificial intelligence literacy curriculum and development tools. In *2021 AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI)*, 2021.

[192] Jessica Van Brummelen, Galit Lukin, and Phoebe Lin. Zhorai. `https://github.com/jessvb/zhorai`, 2021. Accessed: 2022-08-22.

[193] Jessica Van Brummelen, Viktoriya Tabunshchyk, and Tommy Heng. "Alexa, can i program you?": Student perceptions of conversational artificial intelligence before and after programming Alexa. In *Interaction Design and Children*, IDC '21, page 305–313, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384520. doi: 10.1145/3459990.3460730. URL `https://doi.org/10.1145/3459990.3460730`.

[194] Jessica Van Brummelen, Evan Patton, Viktoriya Tabunshchyk, Jeffrey Schiller, Tommy Heng, and Hal Abelson. MIT app inventor Alexa (aka ConvoBlocks). `https://github.com/mit-cml/appinventor-alexa`, 2022. Accessed: 2022-08-22.

[195] Vanessa Van Brummelen. The X-mazing Zorilla, 2018.

[196] Vanessa Van Brummelen. Overtrust and misinformation spread with Alexa and the X-mazing Zorilla, 2022.

[197] Henriikka Vartiainen, Matti Tedre, and Teemu Valtonen. Learning machine learning with very young children: Who is teaching whom? *International Journal of Child-Computer Interaction*, 25:100182, 2020.

[198] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[199] A. Vazhayil, R. Shetty, R. R. Bhavani, and N. Akshay. Focusing on teacher education to introduce AI in schools: Perspectives and illustrative findings. In *2019 IEEE Tenth International Conference on Technology for Education (T4E)*, pages 71–77, 2019. doi: 10.1109/T4E.2019.00021.

[200] Emily K. Vraga, Leticia Bode, and Melissa Tully. Creating news literacy messages to enhance expert corrections of misinformation on twitter. *Communication Research*, 49(2):245–267, 2022. doi: 10.1177/0093650219898094. URL `https://doi.org/10.1177/0093650219898094`.

[201] Marie-Louise Wagner. Ceed: Computational empowerment for emerging technologies in education. `https://cctd.au.dk/projects/ceed-project/`, 2021. Accessed: 2021-09-13.

[202] Xiaoyu Wan, Xiaofei Zhou, Zaiqiao Ye, Chase K Mortensen, and Zhen Bai. SmileyCluster: supporting accessible machine learning in K-12 scientific discovery. In *Proceedings of the Interaction Design and Children Conference*, pages 23–35, 2020.

[203] Zhuxiaona Wei and James A. Landay. Evaluating speech-based smart devices using new usability heuristics. *IEEE Pervasive Computing*, 17(2):84–96, 2018. doi: 10.1109/MPRV.2018.022511249.

[204] Kevin Weng, Jessica Zhu, and Jessica Van Brummelen. CONVO. `https://github.com/jessvb/convo`, 2021. Accessed: 2022-04-21.

[205] Jacqueline M. Kory Westlund, Hae Won Park, Randi Williams, and Cynthia Breazeal. Measuring young children's long-term relationships with social robots. In *Proceedings of the 17th ACM Conference on Interaction Design and Children*, IDC '18, page 207–218, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450351522. doi: 10.1145/3202185.3202732. URL `https://doi.org/10.1145/3202185.3202732`.

[206] Randi Williams, Hae Won Park, and Cynthia Breazeal. A is for artificial intelligence: the impact of artificial intelligence activities on young children's perceptions of robots. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2019.

[207] Randi Williams, Blakeley H. Payne, Daniella DiPaola, Safinah Ali, Phoebe Lin, and Pablo Alejo. How to train your robot companion: MA STEM week with i2 learning. `https://aieducation.mit.edu/i2.html`, 2019. Accessed: 2020-08-28.

[208] Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pages 1–10, 2014.

[209] David Wolber, Harold Abelson, and Mark Friedman. Democratizing computing with app inventor. *GetMobile: Mobile Computing and Communications*, 18(4): 53–58, 2015.

[210] Xizhu Xiao, Porismita Borah, and Yan Su. The dangers of blind trust: Examining the interplay among social media news use, misinformation identification,

and news trust on conspiracy beliefs. *Public Understanding of Science*, 30(8):977–992, 2021. doi: 10.1177/0963662521998025. URL https://doi.org/10.1177/0963662521998025. PMID: 33663279.

[211] Ying Xu and Mark Warschauer. Exploring young children's engagement in joint reading with a conversational agent. In *Proceedings of the Interaction Design and Children Conference*, IDC '20, page 216–228, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379816. doi: 10.1145/3392063.3394417. URL https://doi.org/10.1145/3392063.3394417.

[212] Xi Yang and Marco Aurisicchio. Designing conversational agents: A self-determination theory approach. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. URL https://doi.org/10.1145/3411764.3445445.

[213] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. Gender bias in contextualized word embeddings. In *NAACL-HLT (1)*, 2019.

[214] Xiaofei Zhou, Jessica Van Brummelen, and Phoebe Lin. Designing AI learning experiences for K-12: Emerging works, future opportunities and a design framework. *arXiv preprint arXiv:2009.10228*, 2020.

[215] Jessica Zhu and Jessica Van Brummelen. Teaching students about conversational AI using convo, a conversational programming agent. In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 1–5. IEEE, 2021.

[216] Jessica Zhu and Jessica Van Brummelen. CONVO Appendices 2021. https://gist.github.com/jessvb/7c091e9e8f22e1125b56b1da9d495dc5, 2021.

[217] Naim Zierau, Christian Hildebrand, Anouk Bergner, Francesc Busquet, Anuschka Schmitt, and Jan Marco Leimeister. Voice bots on the frontline: Voice-based interfaces enhance flow-like consumer experiences & boost service outcomes. *Journal of the Academy of Marketing Science*, Jun 2022. ISSN 1552-7824. doi: 10.1007/s11747-022-00868-5. URL https://doi.org/10.1007/s11747-022-00868-5.

[218] Abigail Zimmermann-Niefield, Shawn Polson, Celeste Moreno, and R Benjamin Shapiro. Youth making machine learning models for gesture-controlled interactive media. In *Proceedings of the Interaction Design and Children Conference*, pages 63–74, 2020.