Human-Aided Bots

Pavel Kucherbaev
Alessandro Bozzon
Geert-Jan Houben
Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology

A chatbot is an example of a text-based conversational agent. While natural language understanding and machine learning techniques have advanced rapidly, current fully automated chatbots still struggle to serve their users well. Human intelligence, brought by crowd workers, freelancers, or even full-time employees can be embodied in the chatbot logic to fill the gaps caused by limitations of fully automated solutions. In this paper, we investigate human-aided bots, i.e., bots (including chatbots) using humans in the loop to operate. We survey industrial and academic examples of human-aided bots, discuss their differences and common patterns, and identify open research questions.

The idea of having a conversation with a machine similar to how we converse with a human is not new. In science fiction books and movies, various robots, such as the C-3PO humanoid robot from "Star Wars," and automated personal assistants, such as HAL from "2001: A Space Odyssey," helped heroes in their life and to manage their work duties. The first conversational agents to follow this idea already appeared in the 1960s.[1] Then, it was very hard to program such systems even for a narrow domain; a lot of complex rules were explicitly programmed, as there was no way to quickly and reliably parse user requests to understand what the user wanted. The significant improvements in parallel processing hardware and *natural language understanding* using *deep neural networks*[2] made it easier now to implement such conversational agents. A new market emerged, and major companies compete with their technologies for the leader position.

Messaging applications such as Facebook Messenger and Telegram are widely used by millions of people to interact with friends, colleagues, and companies.[3] Because of the high popularity of such tools, their users are very familiar with their minimalistic interfaces and functionality. Seeking the opportunities brought by modern conversational agents, these messaging applications have started supporting the creation of text-based conversational agents called chatbots, mimicking a conversation with a real human. Companies express commercial interest in such chatbots and already use them in application domains spanning from customer support (e.g., handling returns and replacements at a retail store [https://www.facebook.com/Customer-Support-Bot-1857341381220252/]) to sales (e.g., helping to find and purchase airflight tickets [https://www.facebook.com/TransaviaFlightSearch/]), and team productivity (e.g., organizing SCRUM stand-up meetings in Slack [https://standuply.com/]).

Chatbots implemented using automated techniques, such as rule-based or machine learning algorithms, are still far from being perfect, struggling to serve well user requests and to carry on a meaningful conversation. These issues are especially evident in open conversation domains.[4] In this

Published by the IEEE Computer Society
1089-7801/18/$33.00 ©2018 IEEE

paper, we discuss how human intelligence could be used to address the limitations of fully automated solutions. We discuss different components of the chatbot architecture; we introduce the concept of *human-aided bot,* a chatbot system where at least one architecture component employs human intelligence; we introduce a *reference framework* to discuss human-aided bots and use it to compare existing examples introduced by academia and industry. In this comparison, we consider chatbots where humans intervene during runtime, and we *do not include* chatbots which are only pre-trained on human-generated data. We end this paper with a list of open research questions, aiming to guide and inspire research and industrial communities to take their next actions.

CHATBOT

In Figure 1, we show chatbot architecture components that have been previously introduced.[5] After a chatbot receives a request from a user (e.g., "*What is the weather in San Francisco*"), the *language understanding (LU)* component parses it to infer the user's intention and the associated information (e.g., intent: "*check weather,*" entities: [location: "*San Francisco*"]). When the request is understood, *action execution and information retrieval (AEIR)* takes place, so that the chatbot performs the requested actions or retrieves the information of interest from its *data sources (DS, e.g., gets API response from openweathermap.org for San Francisco)*. Upon retrieval, the *response generation (RG)* component prepares a response to the user (e.g., "*It is +23*°C *and sunny in San Francisco now.*"). A *dialogue management* component is in place to keep and update the context of a conversation (e.g., the current intent, identified entities, or missing entities required to fulfill user requests), to request missing information (e.g., the chatbot asks "For which city would you like the weather forecast?"), to process clarifications by users (e.g., the user replies "*What about tomorrow?*"), and to ask follow-up questions (e.g., the chatbot replies "Would you like as well a forecast for a week?").

Limitations of Chatbots

Each component in Figure 1 is usually implemented using rule-based algorithms or machine-learning models trained with datasets. Unfortunately, such automated approaches are still ineffective in a variety of real-world scenarios leading to a poor performance with:

- *LU* – the interpretation of user requests, due to limited (in size or diversity) training data;[6]
- *DM* – the generation of clarification requests for missing information, due to limitations in dialogue structure programmin;[4]
- AEIR/DS – the retrieval of the requested information—or the execution of the requested action—due to incomplete support for user intents, thus causing the chatbot to fall back to traditional information retrieval techniques (e.g., using search engine[7]) providing users with documents, rather than facts;
- RG – the presentation of the information to the user in a satisfactory fashion—or the generation of inappropriate responses—due to limitations in response templates, question-answer mapping, response synthesis techniques, or associated training data.[8]

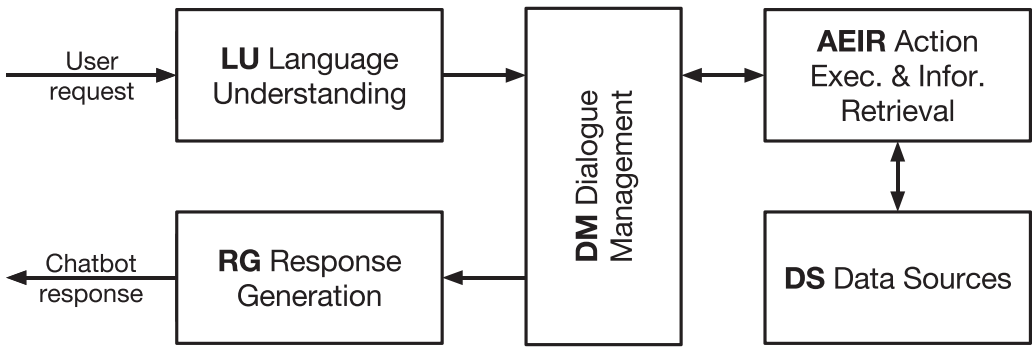

Figure 1. Chatbot architecture.

Intuitively, all the issues above could be easily overcome by a human being proficient with the language used in the conversation and having access to the Web. The computational paradigm that advocates the use of human processing power to solve problems that computers cannot yet solve is called *human computation*.[9]

Human Computation in Chatbots

We refer to chatbots which utilize human computation in at least one component from Figure 1 as *human-aided bots*. Human computation, compared to rule-based algorithms and machine learning, provides more flexibility and robustness as humans can adapt and perform well even when input or instructions themselves change. Still, humans cannot process a given piece of information as fast as a machine, which makes it hard to scale to more user requests.

Human computation is a powerful approach helping to address the challenges of automated approaches. However, in the context of chatbots human computation introduces two extra challenges: *real-time support*—to make sure that users get responses in a reasonable time, and *scalability*—to make sure that even when the number of chatbot users grows, the costs grow only gradually. Below we review several existing human-aided bots, discussing how they combine automated and human computation approaches to serve their users and how they address real-time and scalability challenges.

ANALYSIS OF HUMAN-AIDED BOTS

In our survey, we do not intend to explicitly cover all available human-aided bots (which it is not possible to do in such a publication format). Instead, we target a diversity of solutions and select academic and industrial examples serving different purposes in different domains.

We review 11 human-aided bots in Table 1 using nine dimensions: one describing their purpose and the domain where they function; five dimensions to discuss the implementation of each chatbot component; and three dimensions reflecting the way humans are involved: the source of human workers, how (if any) real-time response is supported, and how scalable the human aid is. We color-code chatbot architecture components reflecting the balance of human computation (red) versus automated (blue) approaches used. To make sure our understanding of selected human-aided bots matches the reality we contacted the authors of academic examples and representatives of companies behind industrial examples to get their feedback.

Purpose and Domain

We distinguish the following purposes that chatbots serve: *informational* (Chorus, Guardian, CRQA, Insurify, SnapTravel, AskWiz, Facebook M), where users obtain information, such as train timetables; *transactional* (Guardian, InstructableCrowd, Legion:Mobile, Calendar.help, Nurtz, Insurify, SnapTravel, Facebook M), where users change the status of another system, e.g., to purchase a train ticket; *conversational (Chorus)*, where users interact with the chatbot for the sake of the conversation itself, such as discussing that trains are late sometimes. Some human-aided bots serve multiple purposes (e.g., Guardian is informational and transactional). We split the domains in which human-aided bots operate as: *generic* (Chorus, Facebook M), i.e., chatbots are ready to answer nearly any user request; *cross-domain* (Guardian, CRQA, AskWiz), i.e., chatbots operate in multiple domains; *domain-specific* (InstuctableCrowd, Legion: Mobile, Calendar.help, Nurtz, Insurify, SnapTravel), i.e., chatbots operate in a narrow domain.

Chatbot Architecture

Most systems rely on human workers at least in some way to understand user requests. While Chorus, InstructableCrowd, and Legion: Mobile rely solely on human workers for language understanding, no system does natural understanding of all user requests completely automatically. AskWiz dynamically dispatches user requests such that their tech support agents are involved only when the automated system fails.

Table 1. Academic and industrial examples of human-aided bots.

			Chatbot Architecture					Human Computation		
	HUMAN AIDED BOT	1. Purpose &	2. Language	3. Dialogue management	4. Action execution /	5. Response generation	6. Data sources	7. Worker	8. Realtime	9. Scalability
Academic	**Chorus – hangouts-based conversational personal assistant, answering generic questions [Huang, 2016a]**	Conversational, Informational / Generic	(2) Human workers	(2) Human workers can see the chat log and facts of the(one) previous session of the current user.	(2) Up to human workers	(2) Human workers: workers propose responses and vote on each other's responses to decide which one to send to the user	(2) Up to human workers	Crowd workers (MTURK)	Task redundancy	Up to MTURK
	Guardian – web-based chatbot which works based on a set of APIs, to which the crowd matches parameters from conversations with users [Huang, 2015].	Informational, Transactional / Cross-domain: API-based	(1) Human workers + machine learning	(1) Dialogue is built around identifying API parameters mediated by human workers	(-2) Up to the API	(1) Human workers clarifying parameters, human workers replying based on Web API response.	(-2) 3rd party(arbitrary API)	Crowd workers (MTURK)	Retainer model	Up to MTURK
	InstructableCrowd – mobile application through which users can create trigger-action rules on their smartphone with a help of the crowd [Huang, 2016b].	Transactional / Domain-specific: Smartphone operations	(2) Human workers	(2) Dialogue is built around defining if/then rules	(-2) Android application based on rules created by human workers	(2) Human workers	N/A	Crowd workers (MTURK)	Task redundancy	Up to MTURK
	Legion:Mobile – mobile application allowing visually impaired people to control their phones with voice cues [Lasecki, 2013].	Transactional / Domain-specific: Smartphone operations	(2) Human workers	(2) Dialogue is build around performing actions the user wants	(2) Human workers performing actions on the user's smartphone via remote desktop	(2) Mediator selecting the action to take based on inputs of other workers	N/A	Crowd workers (MTURK)	Retainer model	Up to MTURK
	CRQA – web-based application to get answers to generic questions with help of the crowd using as well third-party Q/A websites as a knowledge base [Savenkov, 2015].	Informational / Cross-domain: Questions & Answers	N/A	N/A	(0) Relevant answers from web-search; human workers come up with relevant answers;	(0) Trained re-ranking model using workers feedback	(0) 3rd party(Yahoo! Answers, Answers.com, WikiHow.com, websearch), up to human workers	Crowd workers (MTURK)	Retainer model	Up to MTURK
Both	**Microsoft Calendar.help – email-based personal assistant scheduling meetings at the time which fits all the participants [Cranshaw, 2017].**	Transactional / Domain-specific: Scheduling	(-1) Machine learning + human workers(in a form of microtasks and macrotasks)	(-1) Defined workflow + macrotasks managed by human workers	(-2) Scheduling an event	(-1) Machine learning + human workers(in a form of macrotasks)	(-2) user information(calendar)	Crowd (Microsoft crowdsourcing platform, NDA-signed, hourly paid)	Humans working in shifts	Up to the crowdsourcing platform. Escalation from machines to microtasks and later to macrotasks
Industrial	**Nurtz – slack-based assistant proofreading text requested by users with the help of the crowd [http://nurtz.com].**	Transactional / Domain-specific: Writing	N/A	N/A	(2) Human workers proofreading text	N/A	N/A	Freelance remote agents	N/A (average response is in 10 minutes)	Hiring more agents
	Insurify – facebook-messenger based assistant suggesting insurance quotes based on user requests [http://insurify.com].	Informational, Transactional / Domain-specific: Insurance	(-1) Pattern matching + machine learning + human workers	(-1)The user requests and human worker responses then feed back into the machine learning algorithms	(-2) Purchasing insurance	(-1) Machine learning + human workers	(-2) 3rd party(Quotes of supported insurance companies)	3rd party insurance agents	N/A	Hiring more agents
	SnapTravel – facebook-messenger-based assistant helping users to find hotels with the help of travel agents [https://booking.getsnaptravel.com].	Informational, Transactional / Domain-specific: Travel	(-1) Pattern matching + machine learning + human workers	(-1) Machine learning + human workers	(-2) Hotel booking	(-1) Machine learning + human workers	(-2) 3rd party(Offers of supported travel agencies and hotel companies)	Ful-time employees	Humans working in shifts	Hiring more employees
	AskWiz – facebook-messenger-based tech support agent matching users with tech experts [http://dripplar.com].	Informational / Cross-domain: Tech support	(0) Human workers + machine learning	(-1) Human workers answer custom tech questions	(2) Up to human workers	(1) Machine learning + human workers	(2) Up to human workers	Freelance remote agents	N/A	Hiring more agents
	Facebook M – facebook messenger-based personal assistant performing custom tasks with help of a crowd of dedicated employees [http://bit.ly/2qKLaBX].	Informational, Transactional / Generic	(-1) Machine learning + human workers	(-1) Machine learning + human workers	(-1) Custom actions	(-1) Machine learning + human workers	(0) 3rd party + user information	Ful-time employees	Humans working in shifts	Hiring more employees
			N/A	-2	-1	0	1	2		
			Not available	Only machine	Mostly machine, some human	Machines and Humans equal	Mostly human, some machine	Only human		

The way dialogue management is implemented is very similar to language understanding. In academic systems, dialogue management is handled primarily by human workers, while in industrial systems there is usually some predefined dialogue pattern around which conversations are conducted. Two systems (CRQA, Nurtz) do not have any dialogue management support as they focus on atomic request/response interactions.

Chorus, Legion: Mobile, Nurtz, and AskWiz leave AEIR completely up to human workers. More machine-oriented systems perform the following actions: executing an API call (Guardian), executing an Android OS command (InstructableCrowd), scheduling an event in a calendar (Calendar.help), purchasing an insurance plan (Insurify), and booking a hotel (SnapTravel). Systems relying on humans support controlling a user's smartphone via a remote desktop (Legion: Mobile), and proofreading texts (Nurtz). Some systems rely on both machines and humans, such as retrieving answers from web-search and creating new answers with the help of human workers (CRQA), and supporting a wide range of actions (Facebook M) from automatically calling an Uber car to having a human worker to contact Amazon customer support.

Apart from Nurtz, which simply sends to users an edited text with no comments, all other systems rely on humans to generate responses to their users. Academic prototypes Chorus, Guardian, InstrictableCrowd, and Legion: Mobile work such that responses to users are primarily generated by human workers. Other systems rely first on automated ways to generate responses and occasionally on human workers. CRQA ranks answers to pick the one to give using a pretrained ranking model with workers' feedback. Human workers can directly write to the chatbot user (e.g., AskWiz), otherwise, they vote for the response to be sent to the user (e.g., Chorus, CRQA), and in some cases, human workers generate only some responses while others are generated automatically (e.g., Calendar.help).

Three examples (InstructableCrowd, Legion: Mobile, Nurtz) do not have any external data sources. In some systems, the external data sources are completely up to human workers (e.g., Chorus, CRQA, AskWiz). Some systems rely on third-party services to extract information *for* the user (such as Web APIs in Guardian, various Q/A websites in CRQA, quotes of insurance companies in Insurify, offers of supported travel agencies and hotel companies in SnapTravel). Others rely on third-party services to extract information *about* the user (e.g., user calendar in Calendar.help). Not much information is available about Facebook M, but most likely it works as well based on a variety of third-party APIs and data sources.

Human Computation

All academic examples rely on crowdsourcing platforms as a source of human intelligence: Calendar.help is based on a proprietary crowdsourcing platform and others are based on Amazon Mechanical Turk (MTURK). Industrial bots work with remote freelancers (Nurtz, Insurify, AskWiz) and full-time employees (SnapTravel, Facebook M).

Half of all systems ensure real-time responses by keeping workers waiting for tasks to come, as a retainment pool[10] in academic examples, and employees working in shifts in industrial ones. Some chatbots using crowdsourcing platforms simply post redundant requests to attract more workers (as a single task expecting multiple workers to perform it, or as multiple identical tasks) increasing the probability of getting fast responses. Information about how other industrial examples address latency is not publicly available.

The examples relying on crowdsourcing platforms can scale up to the limit of the number of workers (which might be hundreds or thousands) available at the platform at any given moment. Calendar.help has multiple tiers, escalating requests from automated algorithms (e.g., predictions using machine learning algorithms) to human micro-tasks (e.g., structured tasks, to identify meeting time or location) and later to human macro-tasks (e.g., a generic task where a worker needs to make a decision on how to process a given email). Platforms relying on remote agents (e.g., freelancers) scale up by hiring more agents. For downscaling, nothing is needed as human workers only get rewards for their completed tasks. Platforms relying on full-time employees (e.g., such as employees in a call center) need to hire more workers to scale up and lay off or repurpose employees in case of downscale.

DISCUSSION

Below we provide a high-level discussion of the field of human-aided bots, following the same framework we used in the analysis.

Purpose and Domain

Most bots serve informational or transactional purposes. The fact that the single conversational bot Chorus is completely human-based suggests that automated solutions are not yet able to support conversations in the open domain. Having reliability as a priority, most industrial bots operate only in a specific domain.

Chatbot Architecture

The color-coding suggests that academic prototypes heavily rely on the crowd as they are not designed to be used by millions of users: it would cause too many requests to human workers leading to extreme costs. Industrial examples try to manage the costs by relying in the first place on pattern matching and machine learning and escalating to human workers only when automated approaches fail. The single industrial system which relies more on human workers is AskWiz; still, there it is part of the business model, as every request to their human tech experts is expected to be paid for by users. CRQA is a single example where humans and automated approaches work shoulder to shoulder, without either approach predominating.

Human Computation

The source of human workers seems, in general, to correlate with the maturity of the system using it, therefore MTURK is the choice of academic prototypes, freelancers are the choice of startups, and full-time employees is the choice of more established companies such as Facebook. The primary reasons are quality and privacy concerns, since companies as Facebook and Microsoft cannot tolerate poor human inputs in their pipeline. To address this issue, Microsoft relies on proven crowd-workers with whom nondisclosure agreements are signed, and Facebook on full-time employees.

The common approach to ensure real-time responses from human workers is by keeping some workers waiting for tasks to come, which is implemented using a retainment model or scheduled shifts. Redundancy used by Chorus and InstructableCrowd increases the that someone selects the task quickly, but does not ensure it. It is not yet clear how real-time can be ensured with big spikes of request numbers, which is relevant to the issue of scalability in general.

Examples using crowdsourcing platforms seem to scale easily on demand up to a certain limit, examples relying on freelancers scale with some delay caused by finding and recruiting new agents, and the examples relying on full-time employees are the ones struggling to scale the most. Which is one of the reasons Facebook M is currently available only to users in California.

Open Research Questions

We have reviewed and discussed several human-aided bots and now examine the following challenges which are still open for future scientific investigation.

Purpose and Domain

- *Human computation quality control.* Different human computation quality control strategies could be used for chatbots serving different purposes. The current state of the art addresses the collection and verification of information, tasks that are pertinent to *informational* chatbots. Instead, more research is required to understand how to assess human work in *conversational* (e.g., to assure quality in a conversation on a sensitive topic) and *transactional* chatbots (e.g., to assure quality in the task: “call the number and book a table”).
- *Chatbot quality metrics.* Existing chatbot quality metrics focus on measuring how human-like the chatbot is 11. As there are informational and transactional types of chatbots, there is the need for metrics that also account for the quality of the service delivered by the chatbot.
- *Chatbots learning new skills.* Existing informational and transactional chatbots are built with fixed functionality, which it is possible to extend only with coding intervention. Understanding how to design chatbots able *to learn* and *organize* new skills during run-time, could make such chatbots much more powerful and therefore useful. The learning process could be carried in a form of a conversation (with chatbot users or domain experts), micro-tasks (completed by human workers), or even done completely automatically.

Chatbot Architecture

- *Automation versus human labor.* While there are examples of systems with diverse combinations of automated and human work, it is a topic for future investigation how various combinations affect the performance of a human-aided bot and what could be an optimal balance for various domains. In the long term, human involvement is expected to decrease, as automated systems improve in their performance and adaptability.
- *Active Learning from the crowd.* Machine learning models for chatbots are trained during design time. Chatbots can learn proactively, assessing gaps in the training data (using certainty- based, committee-based, or other techniques[12]) and periodically requesting more training data from the crowd, so that the chatbot can serve more user requests automatically in the first place.

Human Computation

- *Scalability and real-time.* Human-aided bots already serve thousands of users (e.g., Facebook M in California). To be able to serve millions of users, the following challenges should be addressed: to ensure that with the growth of the number of chatbot users the costs associated with human computation grow only gradually; to ensure near *real-time* execution of human tasks, even having a big demand of tasks and a moderate supply of workers.
- *Privacy.* Users interacting with human-aided bots often need to share their personal information (potentially also sensitive), which later could be shared with human workers, including those familiar with the user (e.g., a manager asking a chatbot about the best way to fire an employee, and her actual employee happened to be a part-time chatbot human worker who was assigned to answer this request). The topic of privacy is only marginally addressed in the human computation literature, and methods need to be designed and developed to address these discussed privacy concerns.

Addressing these challenges will help to significantly advance the current state of the art in human-aided bots and bring all us closer to the dream of having meaningful and productive interactions with chatbots.

ACKNOWLEDGMENTS

We thank T.-H. Huang (Chorus, Guadian, InstructableCrowd), W. S. Lasecki (Legion:Mobile), D. Savenkov (CRQA), A. Monroy-Hernandez (Calendar.help), T. Kiryazov (Insurify), and H. Fazal (SnapTravel) for their help. This research has been supported in part by the Amsterdam Institute for Advanced Metropolitan Solutions with the AMS Social Bot grant, and by the Dutch national e-infrastructure with the support of SURF Cooperative (grant e-infra17023)

REFERENCES

1. B. F. Green Jr., A. K. Wolf, C. Chomsky, and K. Laughery, “Baseball: An automatic question-answerer,” in *IRE-AIEE-ACM '61*, 1961, pp. 545–549.
2. R. Socher, C. C.-Y. Lin, Y. N. Andrew, and D. M. Christopher, “Parsing natural scenes and natural language with recursive neural networks,” in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 129–136.
3. Business Insider Intelligence, “Messaging apps are now bigger than social networks,” *Business Insider*, 2015. Available at: http://www.businessinsider.com/the-messaging-app-report-2015-11
4. C. Chakrabarti and G. F. Luger, “A framework for simulating and evaluating artificial chatter bot conversations,” in *Proc. 26th Int. Florida Artif. Intell. Res. Soc. Conf.*, 2013, pp. 34–39.
5. M. McTear, Z. Callejas, and D. G. Barres, *The Conversational Interface. Talking to Smart Devices*. New York, NY, USA: Springer, 2016.
6. E. S. AlHagbani and M. B. Khan, “Challenges facing the development of the arabic chatbot,” in *Proc. 1st Int. Workshop Pattern Recogn.*, 2016, 100110Y’16.
7. J. Leber, “Where Siri has trouble hearing, a crowd of humans could help,” in *MIT Technology Review*, 2013. Available at: https://www.technologyreview.com/s/512406/where-siri-has-trouble-hearing-a-crowd-of-humans-could-help/
8. G. Neff and P. Nagy, “Talking to bots: Symbiotic agency and the case of Tay,” *Int. J. Commun.*, vol. 22, no. 60, 2016.
9. E. Law and L. V. Ahn, *Human Computation*. San Rafael, CA, USA: Morgan Claypool Publ., 2011.
10. W. S. Lasecki and J. P. Bigham, “Spoken control of existing mobile interfaces with the crowd,” in *CHI’13 Mobile Accessibility Workshop*, 2013.
11. Z. Yu, Z. Xu, A. W. Black, and A. I. Rudnicky, “Chatbot evaluation and database expansion via crowdsourcing,” in *RE-WOCHAT’16*, 2016, pp. 15–19.
12. G. Tur, R. E. Schapire, and D. Hukkuni-Tiir, “Active learning for spoken language understanding,” in *ICASSP’03,* 2003, pp. 276–279.
13. T.-H. K. Huang, W. S. Lasecki, A. Azaria, and J. P. Bigham, “Is there anything else i can help you with? Challenges in deploying an on-demand crowd-powered conversational agent,” in *HCOMP’16,* 2016, pp. 79–88.

14. T.-H. K. Huang, W. S. Lasecki, and J. P. Bigham, “Guardian: A crowd-powered spoken dialog system for web apis,” in *HCOMP’15,* 2015, pp. 62–71.
15. T.-H. K. Huang, A. Azaria, and J. P. Bigham, “Instructablecrowd: Creating IF-THEN rules via conversations with the crowd” in *CHI EA'16,* 2016, pp. 1555–1562.
16. D. Savenkov and E. Agichtein, “CRQA: Crowd-powered real-time automatic question answering system,” in *HCOMP’16*, 2016, pp. 189–198.
17. J. Cranshaw, *et al*, “Calendar.help: Designing a workflow-based scheduling agent with humans in the loop,” in *CHI’17,* 2017, pp. 2382–2393.

ABOUT THE AUTHORS

Pavel Kucherbaev is a Post-Doctoral Researcher with Web Information Systems Group, Delft University of Technology, Delft, The Netherlands. His research interests include human computation and conversational agents. He received the Ph.D. degree from University of Trento, Trento, Italy. Contact him at p.kucherbaev@tudelft.nl.

Alessandro Bozzon is an Associate Professor with the Web Information Systems Group, Delft University of Technology, Research Fellow with the AMS Amsterdam Institute for Advanced Metropolitan Solutions, and a Faculty Fellow with the IBM Benelux Center of Advanced Studies. His research interests include intersection of crowdsourcing, user modeling, and web information retrieval. Contact him at a.bozzon@tudelft.nl.

Geert-Jan Houben is a Full Professor and the leader of the Web Information Systems Research Group, TU Delft, a Scientific Director of Delft Data Science, a Research Program Leader on Open & Online Education in TU Delft Extension School, and a Principal Investigator in AMS, Amsterdam Institute for Advanced Metropolitan Solutions. His research group covers subjects in the wider field of web engineering and web science, and his research interests include user modeling for web-based systems. Contact him at g.j.p.m.houben@tudelft.nl.