Lifelong and Continual Learning Dialogue Systems: Learning during Conversation

Bing Liu, Sahisnu Mazumder
Department of Computer Science, University of Illinois at Chicago, USA
liub@uic.edu, sahisnumazumder@gmail.com

Abstract

Dialogue systems, also called *chatbots*, are now used in a wide range of applications. However, they still have some major weaknesses. One key weakness is that they are typically trained from manually-labeled data and/or written with handcrafted rules, and their knowledge bases (KBs) are also compiled by human experts. Due to the huge amount of manual effort involved, they are difficult to scale and also tend to produce many errors ought to their limited ability to understand natural language and the limited knowledge in their KBs. Thus, the level of user satisfactory is often low. In this paper, we propose to dramatically improve the situation by endowing the chatbots the ability to continually learn (1) new world knowledge, (2) new language expressions to ground them to actions, and (3) new conversational skills, during conversation by themselves so that as they chat more and more with users, they become more and more knowledgeable and are better and better able to understand diverse natural language expressions and to improve their conversational skills.

Introduction

Building *dialogue systems* capable of conversing with humans in natural language (NL) and understanding human NL instructions is a long-standing goal of AI (Winograd 1972). These systems, also called *chatbots*, have become the front runner of AI advancement due to wide-spread applications such as assisting customers in buying products, booking tickets, reducing stress, and executing actions like controlling house appliances and reporting weather information.

Dialogue systems can be broadly categorized into two main types: **(1) Chit-chat systems** (Shang, Lu, and Li 2015; Li et al. 2016; Serban et al. 2016) designed to engage users and provide mental support by conducting chit-chat type of conversation in a wide range of topics without having a specific goal to complete. **(2) Task-oriented chatbots** (Williams and Young 2007; Wen et al. 2017) designed to assist users to complete tasks based on users' requests, e.g., providing the requested information and taking actions. Most of the popular personal assistants such as Alexa, Siri, Google Home, and Cortana, are task-oriented chatbots. They are primarily designed as Natural Language Interfaces (NLI) that take human NL instructions (commands) and translate them into some actions to be executed by the underlying application. Question-answering (QA) and conversational recommendation systems also fall into this category.

Earlier chatbots were mainly developed with handcrafted rules. With the advent of deep learning, the trend has shifted toward end-to-end conversation modeling (Vinyals and Le 2015; Wen et al. 2017). However, despite their widespread applications, chatbots still have some serious weaknesses: **(1)** A great deal of manual effort is needed to label training data or write rules and compile knowledge bases (KBs). No matter how much data is used to train a chatbot, it is impossible to cover all possible variations of natural language. Thus, when deployed in practice, a well-trained chatbot often performs poorly. **(2)** The pre-compiled KBs cannot cover the rich knowledge needed in practice.

This paper argues that a truly intelligent chatbot should not be limited by its offline-trained model or pre-compiled KB. It should also learn continuously *on the job*, i.e., after model deployment and during conversing or interacting with the (human) end users and thereby, improve its capability over time in a self-motivated and self-supervised manner (Chen and Liu 2018; Liu 2020). Thus, this paper proposes a new paradigm, called **Lifelong INteractive learning in Conversation** (LINC). LINC is like human learning during daily conversations. This paper focuses on three continuous learning capabilities of chatbots: (1) learning factual knowledge in open-ended and information-seeking conversations, (2) learning to ground new NL commands (language expressions), and (3) learning new conversational skills from users. Some initial attempts have been made in (Mazumder et al. 2020a,b, 2019; Hancock et al. 2019; Luo et al. 2019).

A key idea for solving the LINC problem is to exploit the wisdom of the crowd in a multi-user environment (where almost all chatbots operate) to learn new knowledge by asking or interacting with the current user and/or other users to enable the chatbot to learn quickly and effectively. This powerful approach, however, also comes with a shortcoming. The knowledge learned from end-users can be erroneous and some users may even purposely fool the system by providing wrong information or knowledge. We will discuss how to solve this problem to ensure the credibility or trustworthiness of the learned knowledge from the end users. Note, we use the terms: chatbot, bot, agent, NLI systems, dialogue systems interchangeably in the rest of the paper.

LINC: Lifelong Interactive Learning in Conversation

Definition (Lifelong and Continual Learning): It consists of two parts (Liu 2020):

1. ***Formal learning or training*:** At any time point, the learner has learned a sequence of N tasks, $T_1, T_2, \ldots, T_N$ using their corresponding training data $D_1, D_2, \ldots, D_N$. When faced with the $(N+1)^{th}$ task T_{N+1} with its training data D_{N+1}, the learner can transfer the knowledge learned from the previous N tasks to help learn T_{N+1}. If all the tasks are learned in a single neural network, it also needs to solve the *catastrophic forgetting* problem.
2. ***On-the-job learning*:** It refers to learning after the model has been deployed in an application or during model application. Here, the system should (1) discover unknowns and create new learning tasks from the unknowns, (2) collect training or ground-truth data through interactions with users and the environment and through imitation of humans or other AI agents, and (3) incrementally learn the new tasks. The whole process is carried out *on the fly* in a self-motivated and self-supervised manner.

The first part of the definition is taken from (Chen and Liu 2018) based on the early work in (Thrun 1998; Silver, Yang, and Li 2013; Ruvolo and Eaton 2013; Chen and Liu 2014) and the second part of on-the-job learning is proposed in (Liu 2020). Note that we do not use the traditional *online learning* for *on-the-job learning* because in online learning, the agent does not discover unknowns, create new learning tasks, or collect ground-truth training data via interaction with humans and the environment by itself.

LINC is about *on-the-job learning*. During a conversation, the chatbot creates a new learning task T_{N+1} on the fly when it wants to learn a piece of knowledge from a user utterance (e.g., extracting an unknown fact) or encounters a problem (e.g., unable to understand a user utterance or unable to answer a user query).[1] In order to learn the new task T_{N+1}, it needs to acquire the needed ground truth training data D_{N+1}, for which it needs to *formulate a dynamic interaction strategy* $\mathcal{S}$ to interact with the user, e.g., deciding what to ask the user and when to ask the user, and then *execute* $\mathcal{S}$ to acquire the ground truth data. It then incrementally learns task T_{N+1}, which is like the first part of the definition with only one or a few examples. We will not discuss it further in this paper. The key challenge of LINC is how to obtain the ground truth training data on its own initiative.

Obtaining Training Data Automatically

Existing approaches to obtaining the training data is through manual labeling or writing, which is both costly and time-consuming. As chatbots typically work in multi-user environments, we can exploit such environments to obtain the ground truth training data interactively during actual online conversations. This process is both automatic and free.

1. Extracting data or information directly from user utterances (or dialogue history), which can be real-world facts, user preferences, etc. Additional knowledge/data may be inferred from the acquired and existing KB knowledge.

2. Asking the current user when the agent (1) doesn't understand a user utterance, or (2) cannot answer a user query, which forms a new learning task. To obtain the ground truth data, for (1), the agent may ask the current user for clarification, rephrasing, or even demonstration if it is supported (Mazumder et al. 2020b). For (2), it may ask the user for some supporting facts and then infer the query answer (Mazumder et al. 2019, 2020a). In this process, it obtains command-action pairs and question-answer pairs.

3. Asking other users to obtain the answers when the chatbot could not answer a user query. For example, if a user asks "*What is the capital city of the US?*" and the chatbot is unable to answer or infer now, it can try to find a good opportunity to ask another user later "*Hey, do you know what the capital city of the US is?*" If the user gives the answer "*it's Washington DC*," the agent acquires the ground truth (a piece of new knowledge) which can be used in its future conversations or as a piece of training data.

4. Observing user demonstrations, to be detailed later.

Beyond these, the agent may also extract ground-truth data from online documents or online knowledge bases.

Components of a LINC Framework

A typical LINC framework should consist of three major components (although they can be designed as a unified or pipelined framework), as described below.

- **Interaction Module ($\mathcal{I}$).** The interaction module $\mathcal{I}$ is responsible for the modeling of the interaction behavior of the conversational agent, i.e, deciding *when to ask* and *what to ask* the user. Given the current dialogue session (with user), $\mathcal{I}$ dynamically formulates an interaction strategy $\mathcal{S}$ based on the context (e.g., dialogue history, question asked by the user, information acquired from user so far and outstanding information needed to complete a task etc.) and executes $\mathcal{S}$ to carry out multi-turn dialogues for knowledge acquisition (i.e, interactively obtaining the ground truth data from the user). Some of the approaches for designing $\mathcal{I}$ are rule-based dialogue manager (Liu and Mei 2020), finite state machine (FSM) (Mazumder et al. 2020a), reinforcement learning (RL), etc.
- **Task Learner ($\mathcal{M}$).** The task learner module $\mathcal{M}$ solves the current task T_{N+1} created by the agent on the fly. The goal of $\mathcal{M}$ is to build a model to learn from the ground truth data or to capture the semantics of the acquired knowledge or data so that the acquired knowledge together with the past learned knowledge can be used in the agent's current and future learning or dialogue modeling. In dialogue-based factual knowledge learning (Mazumder et al. 2019, 2020a), $\mathcal{M}$ is a factual inference model that is learned to infer new facts from existing (known) facts in the knowledge base (KB) in order to answer questions from the user. In knowledge-grounded conversation modeling (Young et al. 2018), $\mathcal{M}$ can be modeled to learn the embedding of facts (i.e. entities and relations in a KB) and leverage the embedding of contextually relevant facts for knowledge-grounded response generation.

[1] Note that the knowledge learning tasks created by the chatbot for itself to learn are not the same as the tasks that the end-user wants to perform via the chatbot.

- **Knowledge Store ($\mathcal{K}$).** The knowledge store $\mathcal{K}$ is used to store the knowledge (ground truth data) acquired from the user (in an appropriate representation) to be used by $\mathcal{M}$ or $\mathcal{I}$ in current and future conversations. For example, in dialogue-based factual knowledge learning, $\mathcal{K}$ can be a triple store or a knowledge graph storing real-world facts about entities. In knowledge-grounded conversation modeling, $\mathcal{K}$ can be a triple (facts) store or a document store storing web documents or a dictionary representing glossary of various terms and concepts etc.

Design Considerations: Based on the above discussion on the roles of the LINC components, we summarize the design requirements of a typical LINC framework as follows.

1. Real-time knowledge learning and usage. In a given dialogue session, delay in response generation can be annoying to the end user and may result in the discontinuation of the dialogue. Thus, the dynamic interaction strategy formulation by $\mathcal{I}$ and learning of new knowledge by $\mathcal{M}$ should meet the real-time processing requirement so that fast response can be generated within a considerable time.

2. Lifelong and continual learning requirements. The interaction module $\mathcal{I}$ and task learner $\mathcal{M}$ should improve their performance over time as the chatbot engages in dialogues with more and more users and acquires more and more knowledge in the process.

3. Mutual dependency. The design of $\mathcal{K}$, $\mathcal{M}$ and $\mathcal{I}$ are mutually dependent of each other. The data or facts in $\mathcal{K}$ should be stored in formats that can be queried and used in real-time by $\mathcal{I}$ and $\mathcal{M}$. $\mathcal{I}$ should optimize its interaction strategy to acquire the *necessary* data or knowledge from the user that is sufficient to learn a good $\mathcal{M}$. It can be annoying to the user if the agent asks too many questions. Optionally, $\mathcal{I}$ may also track performance of $\mathcal{M}$ over time by gathering some performance statistics which can be used to effectively and dynamically formulate an interaction strategy for optimal knowledge/ground truth data acquisition in dialogues.

In the next three sections, we discuss the specific problems of learning *factual knowledge*, *natural language expressions*, and *conversation skills* during conversation.

Factual Knowledge Learning in Conversation

Many chatbots (e.g., conversational search and question-answering systems) have an explicit knowledge base (KB) storing real-world facts [e.g., (*Chicago*, *CityOf*, *USA*)] (Eric and Manning 2017; Young et al. 2018; Zhou et al. 2018b) to support information-seeking conversations, relevant response generation and help users with product recommendations. One major issue with existing approaches is that the KBs are fixed once the systems are deployed. However, it is almost impossible for the initial KBs to contain all possible knowledge that the user may ask, not to mention that new knowledge appears constantly. It is thus highly desirable for chatbots to acquire new knowledge directly from user utterances or by explicitly asking users questions while in use.

There are many opportunities to learn new knowledge during an actual conversation. Here are a few examples.

- **Extracting facts from user utterances.** For example, while conversing about movies, if the user says "*I watched Forest Gump yesterday. The movie was awesome. Liked Tom Hanks' performance very much.*", the chatbot can extract the new fact (*Forest Gump, isa, movie*) and (*Tom Hanks, performed_in, Forest Gump*) (Liu and Mei 2020). Later, the chatbot can use these facts in future conversations while answering questions like "*Who acted in Forest Gump?*" or generating a response to user's utterance "*I'm feeling bored. Can you recommend a good movie?*".

  The chatbot may even ask the user some related questions (Liu and Mei 2020) to obtain more knowledge. For example, after obtaining (*Forest Gump, isa, movie*), the chatbot may ask a property question: "*What is the genre of Forest Gump?*" If the user answers, then another piece of knowledge is learned. Note that the extraction method proposed in (Liu and Mei 2020) is rule-based, which works with rule-based chatbots. Many deployed chatbots in industry are written with handcrafted rules.
- **Asking questions to learn about unknown entities and concepts.** As unknown entities and concepts appear frequently in user utterances, the chatbot can ask clarification or information seeking questions to the user to acquire facts about new entities or concepts. For example, if the user says "*Is there any good place around for having sushi*?", the chatbot can ask, "*Is sushi a food?*" or "*what is sushi?*". Otsuka et al. (2013) and Ono et al. (2017) have explored the problem of lexical acquisition during dialogues. They proposed approaches where the system makes either implicit or explicit confirmation requests with an unknown term's predicted category and asks the user for verification to acquire the knowledge. However, they did not consider lifelong learning setting.
- **Asking and inferring new facts.** As mentioned earlier, when the chatbot cannot answer an user query, it can ask for some related supporting facts and infer the answer.

  Mazumder et al. (2020a) proposed an interactive learning method called IKAI for learning factual knowledge when the chatbot is unable to answer a user's factual verification (yes/no) question. Here, a fact is a triple, (s, r, t), meaning that entity s and entity t have the *relation* r. Expanding the existing *knowledge base* (KB) by inferring new facts (s, r, t) from existing ones in the KB is called *knowledge base completion* (KBC). Mazumder et al. (2020a) formulated factual knowledge learning as an *open-world knowledge base completion* (OKBC) problem, which allows new s, r, or t to be introduced into the KB. IKAI works with *fact verification* queries (s, r?, t), where ? indicates a query, e.g., (*Obama*, *CitizenOf?*, *USA*) meaning "*Is Obama a citizen of USA?*" IKAI performs two sub-tasks: **(1)** *Interactive acquisition of supporting facts*, formulating an inference strategy to ask the user suitable questions to convert an OKBC query to a KBC query (i.e., making all s, r and t *known* to the KB). This module is implemented using a *Finite State Machine* (FSM). Those user answers (*supporting facts*) are added to the KB. **(2)** *Knowledge inference*, building a predictive model using the supporting facts and the knowledge in the KB to predict if the converted KBC query is true or not.

  In (Mazumder et al. 2019), a complementary problem

was investigated, i.e., when the system is unable to answer a user's WH-question. The system also formulates some questions to acquire supporting facts from the user and then uses these facts and existing knowledge in the KB to answer the question. However, the technique is different.

Although these existing works have explored the above opportunities, several challenges are still not addressed:

1. Understanding context and topic of conversation: An entity or concept appearing in a conversation context, can be ambiguous. E.g., "*apple*" can be a fruit or name of a company. Thus, understanding the topic of conversation or context while grounding facts for response generation is crucial to maintain the relevance of the conversation.

2. Coreference resolution: In multi-turn dialogues, user can often use coreferences to denote an entity or context. Resolving coreferences is important for extracting correct facts and also understanding and answering user's questions.

3. Entity and relation resolution: An entity or relational phrase can appear in various surface forms in user's utterance. E.g. entity "*Obama*" vs "*Barack Obama*" or relation "*born in*" vs. "*place of birth*". Entity and relational phrase resolution and also unseen relation detection are important for optimal knowledge acquisition and inference.

Although (1), (2) and (3) have been studied as independent NLP problems by many existing works, solving them in the conversation modeling context and in lifelong setting and integrating their solutions together to build a *holistic* knowledge learning system remains a challenging task.

Language Learning in Task-oriented Chatbots

Task-oriented chatbots (or commonly known as *virtual assistants*, e.g., Siri, Alexa, and Google Home) are designed as Natural Language Interface (NLI) systems that allow users to issue NL commands to the chatbot and then the system interprets the commands and maps them into some actions to be executed by the underlying application. Existing methods for building NLIs are of two broad categories. The first category **(1)** views the process as end-to-end modeling, where a NL command is provided and the system directly outputs the action to be performed. For example, authors of (Vogel and Jurafsky 2010; Misra, Langford, and Artzi 2017; Tellex et al. 2020) have explored deep learning and reinforcement learning to ground NL commands directly into executable actions. The other category **(2)** focuses on learning a semantic parser to parse the NL command from the user into an intermediate logical form and then, translate the logical form into an executable action in the application (Artzi and Zettlemoyer 2013; Andreas and Klein 2015). In both approaches, the ability to learn previously unknown language expressions and ground them to suitable actions during conversation can greatly improve the performance of NLI systems.

Based on the *various modalities of human-chatbot interactions*, we organize the scope for learning new language expressions in the following two categories:

- **Learning via multi-turn NL dialogues with the user.** In many cases, this is perhaps the most natural way to interact with the end user. For example, say, the user issues the command "*turn off the light in the kitchen*" and the bot fails to execute the intended action. The bot can show/tell a list of *top-k* predicted actions as NL descriptions (as shown below) that can be executed in the current state of the application and asks the user to select the appropriate option from the list.

Bot: Please choose the correct action option below:

option-1. Switch on the light at a given place.
option-2. Change the color of light to a given color.
option-3. Switch off the light at a given place.

The user can easily select the right option (option-3). The action API [say, `SwitchOffLight`(*arg*:place)] corresponding to the selected option (here, option-3) can be regarded as the ground truth for the issued NL command, which is to be used as a new example for learning the new language expression. In subsequent turns of the dialogue session, the agent can also ask additional NL questions and show the option list to acquire the ground truth values of the arguments of the (ground truth) action API.

The *interactive learning method* in (Mazumder et al. 2020b) used a similar technique. The paper in fact proposed a new approach, called ***natural language to natural language*** (**NL2NL**) matching, to build NLIs with the ability to learn to ground user commands continuously. The approach is *application-independent* and requires no pre-collected application-specific training data. Based on the NL2NL idea, a system, called **CML** (*Command Matching and Learning*) was designed to automatically build NLIs for any API-driven applications.

In CML, to build a new NLI (or incrementally add a new task/skill to an existing NLI), the application developer only needs to write a set S_i of *seed commands* (**SCs**) in NL to represent each API action a_i. SCs in S_i are just like paraphrased NL commands from end users to invoke a_i. The only difference is that the objects to be acted upon in each SC are replaced with variables, which are the arguments of action a_i. When the user issues a command C, the system simply matches C with a SC s_k^* of the correct action a^* and in doing so, it also instantiates the variables/arguments for the associated a^* to be executed.

We use the *Microsoft Paint* tool and the API action `drawCircle(`$X1$`,`$X2$`)` (drawing a circle having **color** $X1$ at **coordinate** $X2$) to illustrate the process. Let "*draw a* $X1$ *circle at* $X2$" be a SC for this API, where $X1$ and $X2$ are variables representing the arguments of the API. A user command "*draw a blue circle at (20, 40)*" can be matched or grounded to this SC, where the grounded arguments are $X1$ = '*blue*' and $X2$ = *(20, 40)*.

Since the SCs written by the developer are not likely to cover all possible paraphrased expressions that a user may use to express the same command, CML continually learn new (paraphrased) SCs from users to make it more and more powerful over time.

- **Learning via user demonstrations.** In some cases, NLI systems deployed in practice come with Graphical User Interfaces (GUIs) or remote control facilities to explicitly control devices apart from controlling them via NL

commands. Examples of such systems include task completion robots performing household activities like cleaning robots and personal assistant services integrated with home appliances like Smart TVs, Smart Lights, Smart Speakers, etc. Considering the user has issued an NL command and the bot has failed to execute the intended action, the user may perform the intended action via the GUI or remote control. The bot can record the sequence of executable action(s) performed by the user by accessing the underlying application logs and store the executed APIs as ground truth for the input NL command. The command along with the invoked APIs can serve as labeled examples for learning the command. Related research includes (Forbes et al. 2015; Wang et al. 2017).

Learning of Conversation Skills

A dialogue system can also learn conversation skills to carry out more meaningful and engaging conversations with users. This type of learning is especially important for chit-chat systems so that over time, it can provide more human-like conversation experience to the end users. Some of the major scopes for learning conversation skills are as follows:

- **Learning user behaviours and preferences**: User's dialogue history is a valuable resource to learn each user's behaviours and preferences in various conversation contexts. Given a conversation context, the chatbot can learn whether a user feels more excited or gets annoyed while conversing on a particular topic, what his/her likes and dislikes are etc., to build the user's behavioral and preference profile. The chatbot can then utilize this user profile knowledge in modeling future conversations to make them more engaging with the user. Some related work include (Luo et al. 2019; Liu et al. 2020).
- **Learning emotions and sentiments**: Recognizing emotional state (Zhou et al. 2018a; Pamungkas 2019) and sentiments of the user and leverage it to generate empathetic responses can be useful to building therapeutic chatbots.
- **Modeling situation-aware conversations**: Understanding the situation and spatial-temporal context of a person to decide the conversation strategy is a key characteristic of the human conversation process. Continuously learning from the conversation history of the user provides a scope for chatbots to learn user's conversation profile, e.g., what time of a day the user generally likes to talk or remains busy; understanding spatial-temporal context of the user like whether the user is in a meeting or not, etc. can be useful in building situation-aware proactive chatbots and this can improve user's conversation experience.

Among works on lifelong open-domain conversation modeling, Hancock et al. (2019) proposed a self-feeding chatbot that extracts new training examples (context-response pairs) from the conversations. If the conversation appears to be going well, the user's responses become new training examples to imitate. Otherwise, on making a mistake, it asks the user for feedback to obtain a relevant response. Shuster et al. (2020) designed a role-playing game, where human players converse with agents situated in an open-domain fantasy world and showed that by training agents on in-game conversations with humans, they progressively improve, engaging players for longer durations.

Some Other Challenges

This section highlights some other challenges, which also present potential research opportunities. One obvious challenge is *few-shot learning* as the ground truth training examples obtained during conversations are scarce. But we will not discuss it here as it is already a well-known problem. Below, we focus on a few other major challenges.

1. Dealing with Wrong Knowledge from Users. As we proposed to learn new knowledge through interactions with the end users, one major challenge is how to deal with the issue of acquiring intentional or unintentional wrong knowledge from them. For example, while providing demonstration of an action or in a dialogue session with the agent, the user may perform a wrong action for a given input command or provide an incorrect feedback to the agent to erase its old learning. Then the agent may display unintended behaviour, which might even lead to safety issues.

Since chatbots almost always work in a multi-user environment, such issues can be addressed through a ***cross-verification*** strategy. After acquiring a piece of new knowledge (a new command pattern or action ground truth) in an interaction session, the agent can store these new examples in a unverified knowledge buffer. Next, while interacting with some other users in future sessions to accomplish a related task, it can ask questions to verify the accumulated unverified knowledge. Once a labeled example is verified for K times (from K different random users), the example can be considered as trustworthy and removed from the unverified knowledge buffer to be used in learning or chatting.

2. Revision of Knowledge. Although strategies can be designed to cross-verify any knowledge learned from users, some wrong knowledge will inevitably be learned and stored in the knowledge base. The challenge is how to revise or correct the wrong knowledge once it is detected. This requires a knowledge monitoring system that can detect contradictions in the knowledge base and also a knowledge revision method that can revise the wrong knowledge and also all the consequences inferred from it. These are challenging tasks.

3. Dealing with Safety and Ethical Issues. The ability to learn continuously during conversations comes with the problem of abusive language learning from end users. Also, learning user's behavior, situation and emotional profile and using the knowledge in unintended ways can become a risk to user privacy hacking and biased conversation modeling. Thus, *constrained conversational modeling* is needed to prevent unintended sharing and abusive use of user information.

4. Learning New Task Completion Skills from Users. Modern task-oriented chatbots are deployed with a finite set of task completion skills which they have been pre-programmed with to perform. Building solutions to enable end users to use natural language dialogues to program their own chatbots and endow them with new skills after deployment will lead to personalization of virtual assistants.

Acknowledgments

This work was supported in part by a gift from Northrop Grumman, two grants from NSF (IIS-1910424 and IIS-1838770), and a grant from DARPA (HR001120C0023). We also thank Jiahua Chen, Zhiyuan Chen, Sepideh Esmaeilpour, Geli Fei, Wenpeng Hu, Zixuan Ke, Gyuhak Kim, Huayi Li, Guangyi Lv, Nianzu Ma, Arjun Mukherjee, Qi Qin, Lei Shu, Hao Wang, Mengyu Wang, Shuai Wang, and Hu Xu, for contributing many ideas over the years.

References

Andreas, J.; and Klein, D. 2015. Alignment-Based Compositional Semantics for Instruction Following. In *EMNLP*.

Artzi, Y.; and Zettlemoyer, L. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *TACL* .

Chen, Z.; and Liu, B. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In *ICML*.

Chen, Z.; and Liu, B. 2018. *Lifelong machine learning*. Morgan & Claypool Publishers.

Eric, M.; and Manning, C. D. 2017. Key-value retrieval networks for task-oriented dialogue. In *SIGDIAL*.

Forbes, M.; Rao, R. P.; Zettlemoyer, L.; and Cakmak, M. 2015. Robot programming by demonstration with situated spatial language understanding. In *ICRA*.

Hancock, B.; Bordes, A.; Mazare, P.-E.; and Weston, J. 2019. Learning from Dialogue after Deployment: Feed Yourself, Chatbot! In *ACL*.

Li, J.; Monroe, W.; Ritter, A.; Jurafsky, D.; Galley, M.; and Gao, J. 2016. Deep Reinforcement Learning for Dialogue Generation. In *EMNLP*.

Liu, B. 2020. Learning on the Job: Online Lifelong and Continual Learning. In *AAAI*.

Liu, B.; and Mei, C. 2020. Lifelong Knowledge Learning in Rule-based Dialogue Systems. *arXiv:2011.09811 [cs.AI]* .

Liu, Q.; Chen, Y.; Chen, B.; Lou, J.-G.; Chen, Z.; Zhou, B.; and Zhang, D. 2020. You impress me: Dialogue generation via mutual persona perception. In *ACL*.

Luo, L.; Huang, W.; Zeng, Q.; Nie, Z.; and Sun, X. 2019. Learning personalized end-to-end goal-oriented dialog. In *AAAI*.

Mazumder, S.; Liu, B.; Ma, N.; and Wang, S. 2020a. Continuous and Interactive Factual Knowledge Learning in Verification Dialogues. *NeurIPS-2020 Workshop on Human And Machine in-the-Loop Evaluation and Learning Strategies* .

Mazumder, S.; Liu, B.; Wang, S.; and Esmaeilpour, S. 2020b. An Application-Independent Approach to Building Task-Oriented Chatbots with Interactive Continual Learning. *NeurIPS-2020 Workshop on Human in the Loop Dialogue Systems* .

Mazumder, S.; Liu, B.; Wang, S.; and Ma, N. 2019. Lifelong and Interactive Learning of Factual Knowledge in Dialogues. In *SIGDIAL*.

Misra, D.; Langford, J.; and Artzi, Y. 2017. Mapping Instructions and Visual Observations to Actions with Reinforcement Learning. In *EMNLP*.

Ono, K.; Takeda, R.; Nichols, E.; Nakano, M.; and Komatani, K. 2017. Lexical acquisition through implicit confirmations over multiple dialogues. In *SIGDIAL*.

Otsuka, T.; Komatani, K.; Sato, S.; and Nakano, M. 2013. Generating More Specific Questions for Acquiring Attributes of Unknown Concepts from Users. In *SIGDIAL*.

Pamungkas, E. W. 2019. Emotionally-aware chatbots: A survey. *arXiv preprint arXiv:1906.09774* .

Ruvolo, P.; and Eaton, E. 2013. ELLA: An efficient lifelong learning algorithm. In *ICML*.

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

Shang, L.; Lu, Z.; and Li, H. 2015. Neural Responding Machine for Short-Text Conversation. In *ACL-IJCNLP*.

Shuster, K.; Urbanek, J.; Dinan, E.; Szlam, A.; and Weston, J. 2020. Deploying Lifelong Open-Domain Dialogue Learning. *arXiv preprint arXiv:2008.08076* .

Silver, D. L.; Yang, Q.; and Li, L. 2013. Lifelong machine learning systems: Beyond learning algorithms. In *2013 AAAI spring symposium series*.

Tellex, S.; Gopalan, N.; Kress-Gazit, H.; and Matuszek, C. 2020. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems* .

Thrun, S. 1998. Lifelong learning algorithms. In *Learning to learn*. Springer.

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

Vogel, A.; and Jurafsky, D. 2010. Learning to follow navigational directions. In *ACL*.

Wang, S. I.; Ginn, S.; Liang, P.; and Manning, C. D. 2017. Naturalizing a programming language via interactive learning. In *ACL*.

Wen, T.-H.; Vandyke, D.; Mrkšić, N.; Gasic, M.; Barahona, L. M. R.; Su, P.-H.; Ultes, S.; and Young, S. 2017. A Network-based End-to-End Trainable Task-oriented Dialogue System. In *EACL*.

Williams, J. D.; and Young, S. 2007. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech & Language* .

Winograd, T. 1972. Understanding natural language. *Cognitive psychology* .

Young, T.; Cambria, E.; Chaturvedi, I.; Zhou, H.; Biswas, S.; and Huang, M. 2018. Augmenting end-to-end dialogue systems with commonsense knowledge. In *AAAI*.

Zhou, H.; Huang, M.; Zhang, T.; Zhu, X.; and Liu, B. 2018a. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *AAAI*.

Zhou, H.; Young, T.; Huang, M.; Zhao, H.; Xu, J.; and Zhu, X. 2018b. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*.