A Self-Learning Framework for Large-Scale Conversational AI Systems

Xiaohu Liu, Chenlei Guo, Benjamin Yao, Ruhi Sarikaya
Amazon Alexa AI
Seattle, WA

***Abstract*—In the last decade, conversational artificial intelligence (AI) systems have been widely employed to address people's real-life needs across various different environments and settings. At the same time, users' expectations of these systems have been on the rise as they expect more contextual and personalized interactions with continuous learning systems, akin to their expectation in human-human interactions. Modular systems constructed as pipelines of machine learning models and trained through supervised learning paradigms often struggle to improve user experience due to the less-than-ideal, slow accuracy improvements they undergo, and the privacy concerns associated with manual annotation. Inspired by how humans learn from their experiences and interactions, this article proposes a comprehensive self-learning framework designed to tackle these challenges for large-scale conversational AI systems, fostering continuous automated learning. The proposed self-learning framework comprises three elements: feedback collection, feedback interpretation, and learning mechanisms. Without the need for annotators in the loop, a self-learning conversational AI system autonomously uses a feedback interpreter to subscribe to, interpret, and utilize user feedback to adapt its behaviors through various learning mechanisms. First, the elements of the self-learning framework are described and then applied to Alexa, a large-scale conversational AI system. Subsequently, this article presents its effectiveness in reducing user-perceived defects. Finally, it explores the implications of self-learning for general AI systems and suggests future directions.**

***Index Terms*—Conversational AI, self-learning, user feedback, query rewriting**

I. Introduction

In the past decade, artificial intelligence (AI) has undergone a profound transformation, significantly impacting various aspects of people's lives. Nowadays, users can interact with numerous AI applications to accomplish tasks, access information, communicate with family and friends, and entertain themselves anytime and anywhere. Among these applications, *conversational AI* [1]–[4] has empowered many virtual agents or chat-bot systems, *e.g.,* Alexa, Siri, Google Assistant, Cortana, and Facebook Assistant. At home, users can talk to voice-enabled systems to set alarms and reminders and invoke smart-home functions such as closing blinds, (un)locking doors, and controlling the lights. During their leisure time, users can inquire about recent games and movies or ask general questions. These conversational AI systems were barely imaginable decades ago. However, the rapid technological advancement experienced over the last decade has made them a part of our daily lives.

The unprecedented application and success of conversational AI systems can be attributed to their capability to communicate like humans. Compared to human agents, conversational AI systems excel in terms of availability (accessible anytime and anywhere), knowledge (possessing extensive information beyond that of an average person), and personality (consistently patient, composed, and never forgetful). However, when it comes to learning from past interactions and experiences, AI systems are not nearly as proficient as humans. As demonstrated in Figure 1, AI systems fail to learn from previous defective interactions.

One primary aspect that vastly distinguishes humans of AI systems is humans' natural self-learning abilities, which empower them to autonomously acquire knowledge and understanding. Human learning involves evaluating the existing knowledge, learning from observations and feedback, identifying learning needs, asking questions, and adapting strategies to achieve learning goals. In contrast to the traditional teacher-led learning, self-learning emphasizes learning from firsthand experiences, aligning with the cognitive development theory [5], experiential learning [6]–[8], and the method of learning through experience [9]. According to the cognitive development theory [5], children and adolescents actively construct their comprehension of the world by reconciling the disparities between their understanding and experiences, prompting a reorganization of their mental processes. Experiential learning is a process in which knowledge emerges through transforming experiences [6]. Concrete experiences serve as the foundation for reflection, allowing individuals to assimilate information and develop new insights about the world. The method of learning through experience [9] closely aligns with the experiential learning theory. According to this method, individuals learn best when they interact directly with the world and make connections between their experiences and the concepts they are being taught.

The human-like self-learning paradigm can be widely applied to conversational AI systems. Initially trained through supervised learning, these systems achieve continuous and autonomous improvement by collecting feedback and learning from real-world interactions, resembling the principles of experiential learning.

To improve the response quality of large-scale conversational AI systems from the users' perspective, it is necessary to redesign the traditional system pipeline and introduce new

(TURN 1) USER: *(Agent name), play Fearless.*
SYSTEM: *Playing Fearless (by Pink Floyd).*
(TURN 2) USER: *Stop.*
SYSTEM: ...
(TURN 3) USER: *(Agent name), play Fearless.*
SYSTEM: *Playing Fearless (by Pink Floyd).*
(TURN 4) USER: *Stop.*
SYSTEM: ...
(TURN 5) USER: *Play Fearless by Taylor Swift.*
SYSTEM: *Here's Fearless (by Taylor Swift).*

Fig. 1. An illustrative session between a user and a conversational AI system incapable of self-learning. In Turn 1, the system initiates playback of a song that is different from what the user intended. The user stops the song and repeats the request. However, the system does not learn from the previous error and plays the same incorrect song. Consequently, the user resorts to rephrasing the request by appending the artist's name to it to ensure that the desired song is played.

system components that learn from the user interactions, especially from the previous failures [10].

A conventional architecture [1], [3], [4] usually consists of machine learning (ML)-based components including *Wake Word Detection* (WWD), *Automatic Speech Recognition* (ASR), *Natural Language Understanding* (NLU), *Entity Resolution* (ER), *Dialog Management* (DM), *Skill Routing* (SR), *Natural Language Generation* (NLG), and *Text-to-Speech* (TTS). The WWD component initiates a session when the user invokes the agent's name. In each turn, the ASR component transcribes the user's voice request into textual query (*e.g.,* "*Play Fearless*"). The query is then parsed by the NLU and ER components into the *domain* (*e.g.,* Music), *intent* (*e.g.,* Play Song), and *entity* (*e.g.,* song name *Fearless* and the song ID) information, which are necessary inputs used by the SR components [11], [12] to route the request to the most suitable skill (*i.e.,* back-end application) to fulfill the user's request. Within a skill, after updating the dialog state with the current turn's information, the DM component may either prompt the user for additional details or execute the desired action (*e.g.,* play music). The NLG component generates a textual response, *e.g., "Playing Fearless (by Pink Floyd),"* for the TTS component to vocalize to the user and conclude the turn.

As a pipeline of ML components, a conversational AI system may fail to handle a user's request if any of its components makes an inaccurate prediction. For instance, in Turn 1 of Figure 1, the failure may have originated from the ER component. While the user may have shown their long-term preference in the previous interactions with the system, without a proper feedback loop between these interactions and the system's components, the ER component might disregard the user's preference and fail to resolve the correct song among the numerous music tracks with the same name, "*Fearless*".

A. Annotation Challenges for Large-Scale Conversational AI

ML components for large-scale conversational AI systems are typically built using the supervised learning paradigm, wherein datasets are primarily annotated by data annotators that have been trained to label the expected output for a given input. The target ML model is then trained to learn the prediction function from training data and is expected to generalize its predictions to unseen input cases. For example, to address defective use cases, as depicted in Figure 1, the ER model needs to be retrained with more annotations that involve ambiguous entity names in the input queries and ground truth entities coinciding with the users' past behaviors. However, collecting large annotation sets for every ML component is a challenging and, in some cases, infeasible task. Due to the inherent limitations of manual annotation, large-scale conversational AI systems in the industry are confronted with the following annotation-related challenges.

1) **Speed & Cost**. Increasing the amount of annotated data can significantly benefit supervised ML components. Typically, for each domain, it requires tens of thousands of conversation utterances [13]–[15]. However, due to the limited bandwidth and throughput of individual annotators, it may be necessary to form a sizable annotation team consisting of dozens to hundreds of annotators to collect and annotate a substantial amount of data, ranging from thousands to millions of data samples. Additionally, in conversational systems where ML models encompass multiple components, languages, and locales, separate annotation teams may be required for each specific setup. This amplifies the costs associated with annotation and prolongs the time required for data collection and annotation.
2) **Complexity & Accuracy**. Due to the inherent ambiguity in understanding human language [16], [17], annotators need to know a rich set of contextual and personal conditions in order to understand the user's intent and relevant entities [18]. Oftentimes, annotators can only make guesses based on the limited contextual information available from system logs. For example, when receiving the query *"How is the weather in Olympia?"* an annotator needs to consider the user's location (United States *vs.* Greece) to label the right location entity. To correctly label the entity in "*Fearless*," the annotator must be aware of the device type and its capabilities. The user is more likely to refer to a movie than a song if the request comes from a smart TV remote. If it originates from a smart speaker, it may be challenging for annotators to infer the song without knowing the user's preferences or affinities, and they may have to make assumptions based on the aggregated popularity statistics. Considering the nuanced variations among different conditions, such as distinguishing between Video and Movie, and the extensive catalogs of entities within specific domains (*e.g.,* millions of entities in the Shopping domain), even skilled and well-trained annotators may occasionally make labeling errors. Consequently, this can reduce annotation accuracy and agreement rates among annotators.
3) **Privacy & Ethics**. Large-scale conversational AI systems are designed to serve real users, and conversation sessions may involve sensitive or private information. This raises valid concerns related to privacy

and ethics [19]–[21]. For example, privacy regulations may require that only utterances from consenting users can be used for manual transcriptions where annotators have access to the audio. There may be restrictions on the contextual and personal signals that can be shared with human annotators. This situation has a dual impact on the system. First, it restricts the amount of user data available for annotation, potentially limiting the training data for supervised ML models. Second, annotations from a small fraction of consenting users may only represent part of the user population. If models are trained on such potentially biased data, they may perform poorly, particularly for users who are under-represented in the data.

B. System Challenges for Large-Scale Conversational Systems

Large-scale conversational AI systems are developed to cater to the needs of millions of customers, spanning a wide range of scenarios across multiple languages. This large scale and diversity further intensify the annotation challenges discussed earlier. The sheer magnitude of the user base poses additional system challenges [3], [4], [22].

4) **Misaligned Metrics**. Large-scale conversational AI systems are typically designed as loosely coupled ML components, each built by different teams to facilitate decoupled and parallel development. These components are driven by metrics at the component level. However, it is worth noting that system and component metrics may not always align, and improvements in component-level metrics do not necessarily equate to a better user experience. For example, the NLU component for a specific domain may perform well on in-domain requests for NLU tasks (such as domain and intent classification and named entity recognition), which are measured by NLU metrics. However, this improved NLU component might cause user experience degradation when deployed to the system. The correct slots for the utterance "play dancing on my own tiesto remix" should be {song: "dancing on my own", artist: "tiesto remix"} instead of {song: "dancing on my own tiesto remix"}. If a downstream music provider is expecting the latter, a defective customer experience may ensue when an NLU module is updated with presumably better models.
5) **Slow Learning Systems**. Users have high expectations from AI systems, desiring precise comprehension and engaging responses. However, despite meticulous training, ML models are prone to errors in long-tail use cases. To minimize the duration of unsatisfactory user experiences, conversational AI systems depend on human experts to identify and resolve underlying issues. For example, when trending entities, such as the latest hit song "*Home*," get popular overnight, the system plays other songs called *Home* in response to the request *"Play Home"* due to the commonness of the name; manual interventions may have to be temporarily applied to the system's response. When the system cannot answer queries with common patterns, *e.g.,* queries on cooking recipes, manual analyses are conducted to determine whether the root cause is an NLU intent prediction error, the lack of a vertical recipe search skill, or something else. Regardless of the required actions, manual efforts are needed to trigger and remediate the defect, which often take longer time than desired to deliver an improved user experience. AI systems are expected to learn quickly from user feedback and take alternative actions when defects are detected.

The challenges mentioned earlier are not unique to Alexa and are encountered by various conversational AI systems. In academic research, significant efforts have been made to investigate diverse ML paradigms, such as semi-supervised learning [23]–[28], reinforcement learning [29]–[32], and federated learning [33]–[36], in an attempt to address some, if not all, of the challenges encountered by many conversation AI systems, including Alexa. However, none of these paradigms can simultaneously tackle all the challenges. In Section IV-B, the paradigms' limitations will be examined, specifically in the industry context.

In summary, while conversational AI systems have already made a significant impact on people's lives, there is room for optimization. This necessitates the development of a comprehensive system-wide framework to effectively address the practical challenges mentioned earlier in an industry setting.

II. A Self-learning Framework

In this section, a self-learning framework is proposed, in which a modular conversational AI system can autonomously and continuously improve its accuracy and reduce user defects without human annotators in the loop (see Figure 2). A self-learning framework revolutionizes the process of improving ML models by eliminating the need for component-specific annotation. Instead, it leverages both implicit and explicit user feedback to enhance models. The framework comprises three core elements: feedback collection, feedback interpretation, and learning mechanisms.

Large-scale conversational AI systems are designed to interact directly with users through natural language. During these interactions, users frequently provide implicit and explicit feedback on the systems' responses. Such user feedback is essential to facilitate human-like self-learning. The feedback not only indicates whether the system's responses met the users' expectations but may also suggest what went wrong and how the system should have performed, especially in cases of negative feedback.

In the remaining part of this section, the types of user feedback available in conversational AI systems will be elaborated. The *feedback interpreter* and *learning mechanisms*, the two essential elements that interpret user feedback and improve system behaviors, respectively, to facilitate a self-learning conversational AI system, will also be explained. The steps taken to address all the challenges listed in Sections I-A and I-B with the self-learning framework will then be discussed.

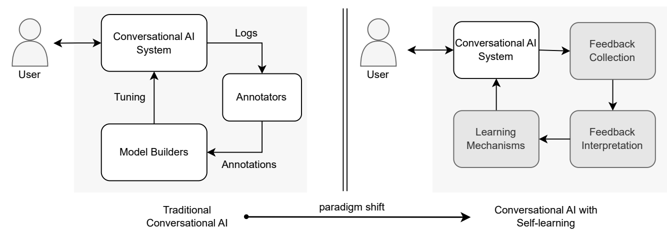


Fig. 2. High-level view of a traditional conversational AI system (left) and its self-learned counterpart (right). The self-learning paradigm leverages both implicit and explicit user feedback to autonomously improve user experience. This paradigm enables the gradual reduction of dependence on manual annotations, which are indispensable in the traditional setting.

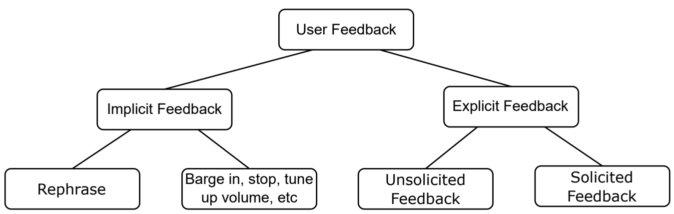


Fig. 3. Various types of user feedback in conversational AI systems.

A. Feedback Collection and Interpreter

As humans, the learning process involves a continuous loop of decision making, taking actions, and collecting feedback from both other individuals and the environment, regarding the outcomes of actions taken. This iterative process enables learning from past experiences, generalizing knowledge, and adapting future behaviors in similar situations. Similarly, in the self-learning process of a conversational AI system, the system receives feedback from users to determine whether to maintain or change its current behavior. On smart devices with conversational AI systems, possible user feedback exists in the users' voice as well as in other multi-modal inputs (*e.g.,* touch and click) as a response to the system's output, which is categorized next.

Types of User Feedback. Figure 3 illustrates the types of feedback relevant to this article. Within a session, the user may provide feedback on a system's response, either explicitly or implicitly. A particular case of implicit feedback is *rephrasing*. When the currently playing "*Fearless*" song is not the one they desired, users may stop it and rephrase their request with more specific instructions, e.g., *"Play Fearless by Taylor Swift."* This type of feedback is particularly important for systems to learn query rewriting, an example of *alternative action generation*, which is further discussed in Section II-B. More implicit feedback can be derived from user behaviors. For example, when the system finds a song that is not what the user wanted, the user may barge in, *i.e.,* interrupt the system while it is responding. In contrast, if the song is what the user requested, they may listen to it without taking any immediate action or turn the volume up to enjoy it. User feedback can also be *explicit*, *i.e.,* directly indicating the system's success or failure. For instance, in *unsolicited* cases, users may offer positive *explicit feedback* such as *"(Agent name), you are smart"* or *"That is helpful, thank you."* They may also express their frustrations, *e.g., "No, not that song."* The system can actively solicit for similar explicit feedback by posing confirmation questions, such as *"Did I play what you wanted?"* after taking actions or *"Did you mean play Fearless (by Pink Floyd)?"* before taking actions.

Feedback Interpreter. Humans engage in self-learning by interpreting feedback to determine whether outcomes are desirable. Similarly, a self-learning conversational AI system should be capable of interpreting user satisfaction or dissatisfaction based on the feedback it receives. To facilitate self-learning, a *feedback interpreter* is needed to translate various user feedback into binary or decimal forms, *e.g.,* user rating and grade, in order to measure user satisfaction with each system response. Throughout this article, the terms *dissatisfaction* and *defect* have been used interchangeably. An interaction is considered defective when the user is dissatisfied. For example, in Figure 1, Turns 1 and 3 should be evaluated as "defective" by the *feedback interpreter* since the user expresses dissatisfaction with the responses.

To comprehensively capture and evaluate the various types of user feedback as described before, the input to a *feedback interpreter* generally consists of system traces and logs, with a focus on two main aspects:

- *System Input/Output*. Users' voice and other multi-modal inputs play a pivotal role in conveying feedback, serving as essential inputs for the *feedback interpreter*. Additionally, other types of system inputs, such as location, date and time, and user profiles associated with a conversation, are also considered implicit contexts for user feedback. Finally, the system's responses should be included in the feedback process. Responses such as *"Sorry, I do not understand"* may imply user dissatisfaction even if the user provides no further feedback on the response.
- *Intermediate Results*. Besides system input and output, the *feedback interpreter* can leverage intermediate results from different ML components. For example, given the ASR transcriptions of users' voice inputs (as in Figure 1), user dissatisfaction is directly indicated by the word *"Stop."* Moreover, with domain/intent results and timestamps from the NLU component, the system can identify implicit user feedback of satisfaction, *e.g.,* the user listens to an album without interruption for an extended period, indicating satisfaction.

B. Learning Mechanisms

Drawing inspiration from human learning mechanisms [5], various types of components were designed in the self-learning framework to adapt the system's behavior based on both successful and failed experiences. The key components include:

- *Alternative Action Generation (Making Corrections)*. By comparing bad experiences to good ones, humans can try alternative actions to obtain a desired outcome. Similarly, AI systems can generate and explore alternative actions. For example, in Figure 1, a self-learning system can learn from the request for "*Fearless*" that the specific user prefers the song *Fearless (by Taylor Swift)* and can thus generate an alternative action to reformulate the request from “*Play Fearless*” to “*Play Fearless by Taylor Swift*” to prevent an unsatisfactory experience in the future. Similarly, the skill-routing module can propose an alternative skill to route to, and the NLU module can generate alternate semantic parses to explore different interpretations of user input.

- *Guardrails (Avoiding Pitfalls)*. Guardrails promptly identify and prevent the system from taking defective alternative actions based on immediate feedback, before the underlying ML models are updated. For example, if a user has repeatedly stopped songs by a particular artist proposed by *Alternative Action Generation*, the system can block the proposed rewrite to prevent future dissatisfaction of the user.
- *Defect Attribution (Identifying the Root Cause)*. Negative experiences can lead humans to ponder where and how things went wrong. When a turn causes user dissatisfaction for a currently supported scenario, at least one component must have made inaccurate predictions, and user feedback may suggest where those inaccuracies have occurred [37]. For example, the ambiguity in the request “*Fearless*” as shown in Figure 1, suggests that the Entity Recognition (ER) component failed to align with the user’s initial expectations. Consequently, a mechanism that attributes this negative experience to the ER component, while providing *synthetic* annotations, can be developed.
- *Retraining (Making Generalizations)*. Humans are adept at making effective generalizations from limited data. Traditional ML systems also excel at making generalizations from limited labeled data. In a self-learning conversational AI system, retraining is regularly conducted at a fixed cadence (*e.g.,* weekly) or automatically triggered when the defect attribution module accumulates a substantial amount of synthetic training data.
- *Consulting (Requesting Clarification)*. Humans recognize that in situations characterized by ambiguity or a lack of information, negative experiences may occur. Similarly, conversational AI systems can include a consulting component responsible for involving users or skill developers and requesting their knowledge and intervention in the system’s behavior. This component can seek confirmation from users regarding specific actions or solicit feedback on the system’s responses, as shown in Figure 3. When the system detects a surge in unsatisfactory user requests for a particular Alexa skill, the consultation component may consult skill developers to optimize the skill for more effective handling of those requests.

C. Addressing Annotation and System Challenges

This section presents a description of how a large-scale conversational AI system can tackle the challenges discussed in Sections I-A and I-B, using the aforementioned self-learning framework. Figure 2 shows a high-level transition from the traditional conversational AI system to a self-learning version. In the traditional loop, annotators sample and annotate logs to create annotation datasets. Modelers use these annotations

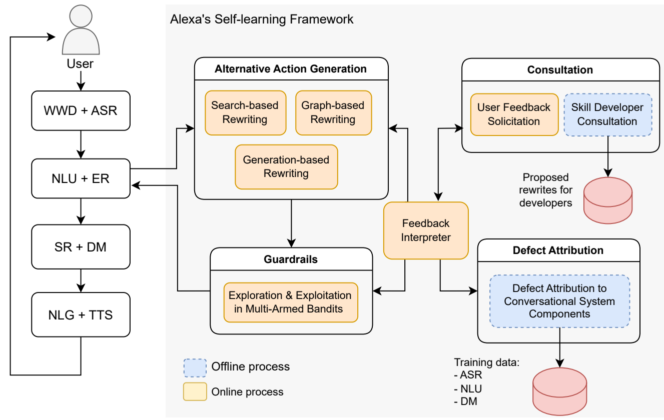


Fig. 4. An overview of the proposed self-learning framework. Using query rewriting as an example, the feedback collection via consultation, the feedback interpreter, and the learning mechanisms, *i.e.,* alternative action generation, guardrails, and defect attribution, are shown in the figure. The defect attribution component provides training data for the conversational AI components to improve the identified errors. The skill developer consultation component is responsible for storing the suggested rewrites, which are subsequently reviewed by modelers, facilitating the ongoing refinement of the system.

to build better ML models to fine-tune the conversational AI system and enhance user satisfaction.

In the self-learning paradigm, user feedback is collected or solicited during regular conversations. The feedback is interpreted to inform the self-learning components (*i.e.,* alternative action generation, guardrails, defect attribution, retraining, and consulting) to automatically adjust the system’s behavior.

In contrast, the self-learning framework benefits from a significant volume of user interactions, amounting to millions of daily interactions. This abundance of user interactions provides valuable signals that can effectively reduce the reliance on manual annotation for model improvement. As a result, it addresses the challenge of speed & cost (Challenge 1). Since system improvements are realized with user feedback (see Figure 3), a higher accuracy can be achieved, effectively addressing the challenge of complexity & accuracy (Challenge 2). The challenge of privacy & ethics (Challenge 3) is mitigated as the system relies on an automatic feedback interpreter instead of human annotation. Moreover, since the user feedback-based learning mechanism focuses on end-to-end system behavior rather than individual component-level metrics, Challenge 4 is addressed. With the defect attribution mechanism in place, feedback becomes automatically actionable, targeting specific components. Finally, the self-learning framework substantially reduces the need for slow and ad hoc manual efforts, thereby alleviating the system of Challenge 5 and providing more rapid and efficient learning mechanisms.

III. Self-learning Conversational AI at Alexa

To illustrate the self-learning components in Alexa, query rewriting was specifically highlighted as a key scenario. While prior research has explored various core self-learning components [38]–[42], this study is the first to describe the overall framework and interconnect all the components, thereby providing a comprehensive perspective. The proposed self-learning framework includes both online and offline components and processes, as depicted in Figure 4.

The feedback interpreter, as discussed in Section III-A, plays a pivotal role in determining whether a dialog turn is defective, a critical function that significantly impacts the performance of all other components. The alternative action generation component utilizes various query rewriting techniques to propose rewrites, as outlined in Section III-B. It is important to note that this component represents only one type of the alternative action generator, with several others available, such as ASR n-best, NLU/ER n-best, and more. To ensure a seamless user experience, the guardrail component assesses the potential impact of the proposed rewrites to prevent defective rewrites from reaching the NLU component (Section III-C). Additionally, in this section, Algorithm 1 outlines the process of interconnecting all online components. The feedback consultation process comprises two steps: first, an online feedback collection phase involving Alexa users and an offline confirmation with Alexa skill developers, as detailed in Section III-D. Moreover, the defect attribution component identifies the elements outside the self-learning framework that contributed to the identified errors. It provides valuable training data for improving specific ML models (Section III-E).

A. *Feedback Interpreter*

Interaction turns and sessions represent the finest-grained units within conversations. Given that 1) the Alexa pipeline in Figure 4 is invoked during each turn, and 2) feedback indicating users' (dis)satisfaction may lie in *other* turns in the same session as illustrated in Figure 1, Alexa's feedback interpreter was designed to perform session-aware, turn-level defect predictions. More specifically, the feedback interpreter is responsible for taking a fully completed session and interpreting the user satisfaction of each turn through explicit (*e.g.,* Alexa confirming the original request) or implicit feedback (*e.g.,* a follow-up request rephrasing the previous one), in addition to examining the conversation history, including queries and responses. For instance, in Turns 1 and 3 in Figure 1, the system played the song *Fearless* by a different artist from what the user had in mind. The feedback interpreter should predict that Turns 1 and 3 are defective.

Given a session of n turns, the Alexa feedback interpreter [39], [40] makes one prediction per turn, with the session as the context. To facilitate this process, all queries and responses in each session were concatenated in chronological order [CLS][USER]q_1[SYS]r_1[USER]$q_2 \ldots$, starting with a special token [CLS] and separated by two special tokens [USER] and [SYS] to distinguish user queries and system responses, respectively. The terms q_i and r_i represent the query and response, respectively, at turn i. To account for the temporal gap between turns, the temporal difference of each turn relative to the target turn t was computed, and bucketed into multiple bins. The special bin labeled 0 corresponded to the current turn t, while negative bin values were assigned to turns preceding t, and positive values denote turns succeeding t. The sequence of tokens was passed through an embedding layer to generate

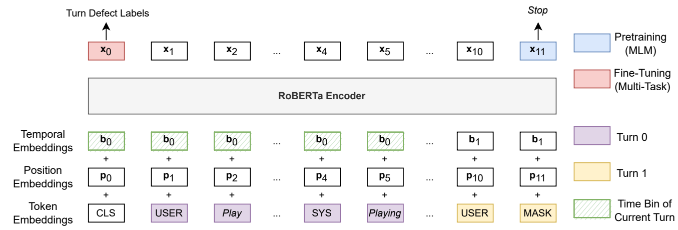


Fig. 5. The architecture of the Alexa feedback interpreter model. Only one follow-up turn is shown, denoted as b_1 for the purpose of illustration in the figure. Two turns were used before and after the current turn to capture the session context in Alexa.

an embedding sequence for the entire session and summed with the sequence's corresponding temporal bin and position embeddings. The resulting representation was then fed into the RoBERTa [43] model, from which the [CLS] output embedding was used as the representation of the turn t to be classified. A comprehensive view of the model's architecture is illustrated in Figure 5.

The training of the feedback interpreter model consisted of two stages: pretraining and fine-tuning. In the initial pretraining phase, the model was exposed to millions of dialog sessions spanning several weeks, using the masked language modeling approach. Subsequently, the model was fine-tuned using manual annotations sourced from user satisfaction datasets [39]. It is worth noting that the patterns of user feedback remained relatively stable over time, so the feedback interpreter did not need frequent retraining. This stability in training data ensured that Alexa's self-learning process remained continuous and efficient, as illustrated in Figure 2.

B. *Graph-, Search-, and Generation-Based Query Rewriting*

Query rewriting is a key application of the alternative action generation. When a query q is issued, resulting in a rewrite q', there are three possible scenarios to consider: 1) a successful rephrase $\langle q, q' \rangle$ has been detected in the past; 2) this rephrase has not been detected before, but q' was previously issued and resulted in user satisfaction; 3) and q' was never issued to the system. To handle these scenarios, different approaches were employed. For the first scenario, a *graph-based* [38], [44] model was employed to leverage known rephrases. For the second scenario, a *search-based* [45], [46] approach was applied to provide similar rewrites. For the third scenario, a generation-based query rewriting method was implemented to overcome the absence of rephrases.

Graph-Based Rewriting. With a substantial collection of successful rephrases (tens of thousands of them), denoted as $R = \langle q, q' \rangle$, available, along with appropriate semantic normalization applied to the queries, a straightforward approach would automatically rewrite query q as q' whenever q is encountered. However, considering that the same q can be rephrased into different q's in R, and there might be rephrasing attempts by other users on q', it raises the question of how the system chooses an optimal q' based on all the available information in R. To answer this question, the rephrase chains

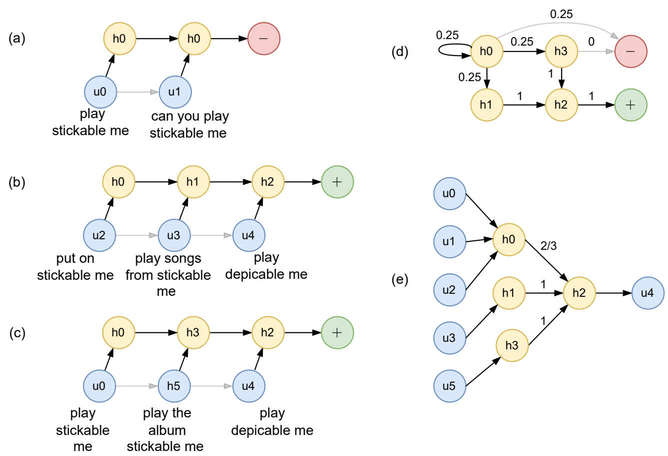


Fig. 6. An illustrative example of a Markov-based rewriting graph. Utterances are represented by u, and the hidden states h are the following entities: $h_0 \rightarrow$ "Song Name: stickable me"; $h_1 \rightarrow$ "Song Name: stickable me | Media Type: songs"; $h_2 \rightarrow$ "Album Name: despicable me"; $h_3 \rightarrow$ "Album Name: stickable me | Media Type: songs". The $+$ and $-$ nodes refer to success and failure, respectively. Recurring user rephrases such as (a), (b), (c) are encoded in absorbing Markov chains. By resolving the Markov model, as in (d), using the transition weights between the hidden states, the rewrite u_4 that is more likely to result in a successful query as in (e) is surfaced.

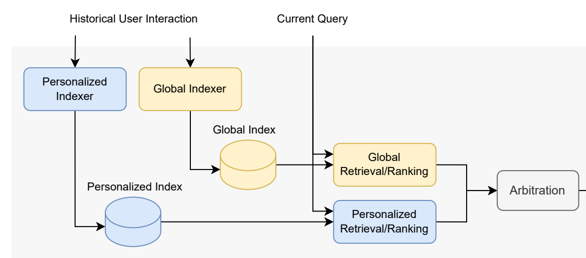


Fig. 7. An overview of the search-based rewriting component. The system leverages past user interactions to build personalized and global (*i.e.,* across users) indices. Based on their semantic similarities, the global and personalized rankers retrieve the top rewrites, which are arbitrated to determine the final rewrite. The image was sourced and modified from [45].

in R were encoded into an absorbing Markov model to surface rewrites via collaborative filtering [38]. Specifically, a Markov graph was constructed, where the nodes consisted of all queries and two special ones, '+' and '-', indicating whether a turn is satisfactory or not (see example in Figure 6). The edges were formed by connecting rephrase pairs $\langle q, q' \rangle \in R$ and then transition probabilities were associated with the edges using the statistics in R. Query q was rewritten as q', which is reachable from q to maximize the probability of a random walk starting from q and arriving at node '+' via q'. In the case of $q' = q$, indicating that the original query is already successful, no rewrite was performed.

Search-Based Rewriting. To generalize the knowledge from the collection R to queries q without previous successful rephrases and to personalize different users' results, *search-based* rewriting approach was devised to complement the graph-based rewriting. This approach addresses common mistakes across users (*e.g., "tooth or dare"* $\rightarrow$ *"truth or dare"*) and user-specific ambiguities (*e.g.,* by *"play imagine"* some users may refer to the artist John Lennon, while others may mean Ariana Grande). As depicted in Figure 7, two indices were created: a *global* index containing rewrites for all users and a *personalized* index per user. Queries were encoded as embeddings via both the Deep Structured Semantic Model [47] and a CNN-based variant [48], where the encoders were trained with Approximate Nearest Neighbor Negative Contrastive Learning [46]. With the query q issued, candidates were retrieved from the indices via fast vector similarity search [49] for candidate queries q' with similar semantics. Next, candidates from both the global and personalized indices were ranked. Because the retrieval stage narrowed down the candidates, the candidate ranking stage was able to leverage more complex features and models to score the surviving candidates. Specifically, an attention-based [50] neural feature extractor was trained to compare a candidate q' with the query q in a fine-grained manner. The comparison score was then fed into a ranking model, together with aggregated information on q' in the indices and other shallow features. It is crucial to have a personalized index, as rewrites can vary from person to person. For example, the utterance *"how is the weather in Wilkeson?"* may be rewritten as *"how is the weather in Wilkeson, Washington?"* using a global index, but it should be *"how is the weather in Wilkerson, California?"* for a customer in California. Finally, the top-ranking results from both the global and personalized indices were arbitrated by prioritizing the personalized ones if both ranking scores exceeded their predefined thresholds.

Generation-Based Rewriting. In both the graph-based and search-based rewriting approaches, the rewrites were selected from previous queries that resulted in user satisfaction. However, these approaches can be restrictive when the system has not previously encountered ground truth rewrites before, especially for less-common utterances observed in domains such as Video, Shopping, and question-answering (*a.k.a.* referred to as Knowledge domain). To overcome this limitation, a generation-based rewriting approach [41], [42] was introduced to complement the other two rewriting methods. The problem was framed as Seq2Seq [51] tasks at both the query and entity levels. At the query level, the successful rephrases $R = \{\langle q, q' \rangle\}$ were used to fine-tune the BART model [52] so that the model could learn from rewrites such as *"Turn on Moonlight Sonata"* $\rightarrow$ *"Play Moonlight Sonata."* At the entity level, an entity correction model was additionally trained to learn from a subset of successful rephrases where the rephrase only changed the entity names, *e.g., "Who sang Staring to Sun?"* $\rightarrow$ *"Who sang Staring at the Sun?"* The entity correction model is also a Seq2Seq model, but the output is constrained to real entity names using trie-based decoding to avoid generating invalid entities. Both models enable the system to generate rewrites that have never been uttered by users before, thereby increasing the system's recall of alternative rewrite generation.

C. Guardrails via Online Exploration

A model that relies solely on historical rephrases to conduct rewrites may face challenges in adapting quickly to the ever-changing dynamics of the real world. For example, based on past sessions such as that in Figure 1, the system may have previously rewritten incoming queries such as *"Play Fearless"* as *"Play Fearless by Taylor Swift."* However, since *"Fearless"* is a common song name, a new hit song with the same name by a different artist can gain popularity overnight, making the previous rewrite outdated. While the feedback interpreter can promptly detect defects, updating the query rewriting models might take hours to days. To bridge this gap, a guardrail mechanism is devised using *online exploration*. This mechanism is versatile and can explore alternative hypotheses, such as different ASR transcriptions, alternative NLU semantic parsing results, and handling ambiguous entities.

As an example of rewrite exploration, Algorithm 1 is applied to each dialog turn to select an action. First, a multi-armed bandit approach [53]–[57] is adopted to balance exploration and exploitation. A set A is then initialized to keep track of all possible actions (arms), including the original query and feedback solicitation actions (Lines 3 and 4). Rewrite providers generate alternative rewrites, which are then added to the set A for exploration (Lines 5-8). Subsequently, samples from the arm's posterior Beta distribution are taken in Line 11, and the arm with the highest sampled value (Line 13) is chosen following the Thompson sampling method [55]. The selected arm can represent the original query, a feedback solicitation, or a rewrite, providing a flexible and adaptive approach to the exploration process. Next, the Beta distribution parameters for the arms chosen in the previous T turns are updated (in our experiments, T is set to 2). Based on delayed rewards from the feedback interpreter, α is increased if the action is not defective; otherwise, β is increased (Lines 14-20).

Algorithm 1 A self-learning algorithm

1: **Input** q: the input user query, P: a set of rewrite providers including search-based, graph-based, and generation-based providers.
2: **Output** an action: the original query, a rewrite, or a feedback solicitation action
3: $c \leftarrow$ feedback solicitation actions
4: Initialize a set to include the original query and feedback solicitation actions: $A \leftarrow \{q, c\}$
5: **for** each provider $p_i \in P$ **do**
6:     $q' \leftarrow p_i.generate\_rewrite(q)$
7:     Add q' to set A
8: **end for**
9: **for** each arm $a_i, i = 1, 2, \ldots, size(A)$ **do**
10:     Initialize the Beta distribution parameters if not done previously: $\alpha_{a_i} \leftarrow 1$ and $\beta_{a_i} \leftarrow 1$
11:     Sample from the arm's posterior Beta distribution: $r_i \sim \text{Beta}(\alpha_{a_i}, \beta_{a_i})$
12: **end for**
13: Choose the action a_i with the highest sampled value: $i = \arg\max_i r_i$
14: **for** each selected action a_t at previous turns $t = -T, -T+1, \ldots, -1$ **do**
15:     **if** $feedback\_interpreter.is\_not\_defect(a_t)$ **then**
16:         $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$
17:     **else**
18:         $\beta_{a_t} \leftarrow \beta_{a_t} + 1$
19:     **end if**
20: **end for**

D. User and Skill Developer Consultation

When the feedback interpreter detects potential defects, Alexa generates alternative actions, *i.e.,* rewriting a request, to explore and recover the experience or consults users and skill developers to confirm the rewrites.

User Feedback Solicitation. Alexa solicits feedback either before or after handling a request [58], referred to as pre-action and post-action feedback solicitation. For pre-action solicitation, the user feedback solicitation sub-component first confirms the rewrite with the user through explicit consultation. For example, if the system rewrites *"turn on Staring to Sun"* to *"play Staring at the Sun"*, it may ask the user, *"Did you mean play Staring at the Sun?"*. If the user verbally confirms, the system marks the rewrite as a positive user experience and executes the action, *e.g.,* plays the song for the user. Otherwise, the rewrite is retained as a negative sample for future references, potentially for retraining purposes. In contrast, for post-action solicitation, when Alexa has low confidence in addressing the user request, the feedback solicitation sub-component elicits feedback after the interaction has taken place. For instance, it may ask, *"Did that answer your question?"*.

Skill Developer Consultation. Alexa offers a wide range of skills, with many targeting long-tail user requests. While built-in skills, *e.g.,* Music, Video, Alarms, and Shopping are well established and continually optimized, long-tail requests often lack sufficient user feedback due to their infrequent occurrence. As a result, user dissatisfaction may not be effectively addressed by fully automatic learning mechanisms, particularly when the feedback interpreter makes less confident predictions. The skill developer consultation component addresses this by using the feedback interpreter to identify queries that frequently result in user dissatisfaction and suggests rewrites that may match the users' intent based on past skill usage. Modelers can then leverage this consulting mechanism by confirming whether a suggested rewrite is better (e.g., modelers can accept *"play thunderstorm sound"* as a rewrite of *"play thunderstorm sleep sound"*) or provide valid rewrites by themselves. Once those rewrites are approved, they remain monitored by the feedback interpreter, allowing for ongoing evaluation and improvement.

E. Defect Attribution

As described in Section III-B, when the feedback interpreter detects defects, Alexa explores alternative actions, such as rewriting user queries. However, this approach addresses defects one utterance at a time. To generalize learning and avoid similar defects in the future, feedback is used to improve the underlying ML models. Previously, a Failure Point Isolation (FPI) model was developed to attribute feedback to a particular system component [37]. A similar mechanism was also implemented for NLU component [59].

(TURN 1) USER: *Play it-will rain.*
[*ASR output:* Play evil rain.]
SYSTEM: *Cannot find the song evil rain.*

(TURN 2) USER: *Play it will rain.*
[*ASR output:* Play it will rain.]
SYSTEM: *Playing it will rain by Bruno Mars...*

Fig. 8. A sample session for ASR defect attribution. The song *"it will rain"* gets mistakenly transcribed as *"evil rain"* because the user pronounced "it will" closely together at a fast pace.

In this article, ASR defect attribution is used to illustrate the defect attribution method. When the ASR component causes user dissatisfaction in a conversation turn, user feedback often exhibits particular patterns that suggest the ASR component is responsible. For example, in Figure 8, when the user says *"Play it will rain,"* the ASR component may incorrectly transcribe it as *"Play evil rain,"*, leading to an unresolved entity and a response *"Sorry, I cannot find the song evil rain."* In the subsequent turn, the user may realize that her voice was not clearly heard and may repeat the query with *"it will"* emphasized. In such case, the acoustic similarity between the ASR outputs of the first and second turns, along with the user's implicit feedback of repetition and emphasis, strongly indicate that the ASR is at fault. Consequently, this input/output pair can be used as a negative sample in future ASR model retraining.

Formally, given a pair of utterance u and its ASR transcription q in a session, ASR defect attribution aims to decide whether q is a correct transcription denoted as $\hat{q}$, using contextual information in the session including implicit and explicit user feedback. The ASR defect attribution model was implemented using a similar architecture as the feedback interpreter, where the input includes the user's speech u, represented using wav2vec 2.0 [60], and the output label indicates whether the ASR output q matches the correct transcription $\hat{q}$.

F. Empirical Results

Online experiments were conducted to assess the functionality and impact of the proposed self-learning solution on the Alexa user experience. First, it is shown that the feedback interpreter is consistent with humans in measuring user experiences. For brevity, the experimental study did not include all the self-learning components. Instead, the end-to-end effectiveness of the self-learning framework, which consistently improves users' experiences over time, was demonstrated.

Experimental Methodology. The experiments were conducted using an A/B test methodology [61]. Among the stream of queries flowing through Alexa, the query rewriting methods used in these experiments selectively decide whether to rewrite based on their confidence. When the decision to rewrite was made, the experiment continued with a probability of 98%. In the remaining 2% of cases, rewriting was intentionally avoided. This "hold out" design enabled a fair comparison between Alexa traffic with and without query rewrites, as measured by the feedback interpreter.

Feedback Interpreter Quality. To evaluate the performance of the feedback interpreter, a total of 20K turns were randomly sampled and annotated with personally identifiable information programmatically removed. Figure 9 presents the defect reduction rates measured by human annotators and the feedback interpreter for turns with and without rewrites. It is important to note that these annotations were not used to train the feedback interpreter but rather for its assessment. The feedback interpreter is the most critical component of the self-learning framework. It is important to periodically monitor its performance to ensure that it does not deteriorate over time. This evaluation was conducted with a relatively small annotation effort to confirm its continued effectiveness.

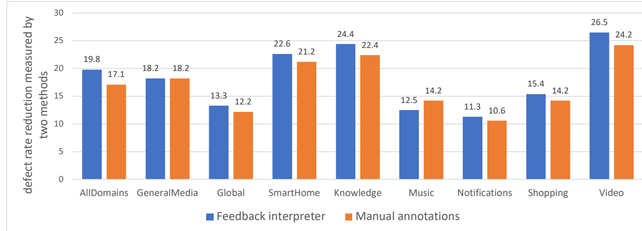
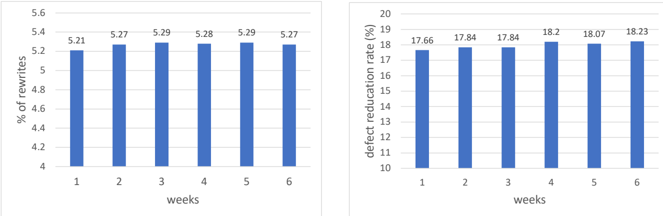


Fig. 9. A comparison of defect rate reduction measured by the feedback interpreter and manual annotations on a 20K-request sample of Alexa traffic. The feedback interpreter estimated that the self-learning framework reduced defects by 19.8%, while manual annotations estimated an overall reduction of 17.1% in defects, calculated as the average reduction across domains (*i.e.,* represented by the "AllDomains" bar). This comparison demonstrates that the feedback interpreter performs on par with the manual annotations, effectively eliminating the need for human-in-the-loop dependencies in defect detection.

The overall defect rate reductions and domain-specific breakdowns in eight domains, including GeneralMedia, SmartHome (*e.g.,* voice-controlled lights or security cameras), Music, Notifications (setting timers and alarms), and Knowledge, *etc.* are reported herein. It was observed that the defect rate reduction measured by the feedback interpreter (blue bars) closely aligned with that measured by manual annotation (orange bars), with only a 2.7% absolute difference across all domains. This suggests that the feedback interpreter is a reliable alternative to manual annotation for assessing conversations. The most significant discrepancy was observed in the Knowledge, Video, and Shopping domains. Annotating conversations in these domains is more challenging than in the others. Users can ask Alexa open-domain questions in the Knowledge domain, while the Video and Shopping catalogs are considerably large. Human annotators often lack the required knowledge and need to search the web to understand and annotate conversation sessions.

The feedback interpreter component does not rely on conventional feature engineering techniques to classify feedback into different types. However, it is still valuable to observe the frequency of each feedback type in the proposed system. An analysis reveals that in cases where a turn in the conversation was defective, the majority of the feedback was implicit, accounting for approximately 93% of the feedback. Unsolicited feedback comprised around 2.5% of the total feedback, while solicited feedback accounted for 4.5%.

Impact of Query Rewriting. The graph-based models [38] were deployed across 15 locales spanning six languages, and an online A/B testing was conducted. The results showed a significant (p-value $\leq$ 0.0001) defect reduction, with a rela-



(a) Percentage of rewritten requests.

(b) Defect rate reduction between the treatment and control requests.

Fig. 10. The online rewriting performance for a randomly selected six-week period. Figure (a) illustrates that the self-learning framework achieved a rewriting rate of 5.2-5.3% for utterances during the six-week evaluation period. Figure (b) displays the reduction in defect rates for two groups of requests: treatment and control. In the sixth week, for example, the treatment requests exhibited a defect reduction rate of 18.23% relative to the control requests.

TABLE I
QUERY REWRITE EXAMPLES

No.	ASR outputs	Query rewrites
1	turn off that door	turn off back door
2	play fever for me	play feeder for me
3	play chef by Juice World	play chimp by Juice WRLD
4	play Brut by Flurry	play Breathe by Fleurie
5	play no doubt spiderwebs	play spiderwebs by No Doubt
6	play by Olivia Olivia Rodrigo	play music from Olivia Rodrigo
7	play I love rock and roll	play I love rock and roll by Britney Spears
8	play hello	play hello by Adele
9	play a. b. c.	play the alphabet song
10	turn the volume to half	volume five

tive reduction ranging from 22.73% to 31.22%. The search-based rewriting model [45] was deployed, and online A/B experiments also demonstrated a significant (p-value ≤ 0.001) relative defect reduction rate of 13%.

Furthermore, the overall impact of all rewriting methods applied over time was evaluated, focusing on how they change the user experience at scale. Figure 10(a) shows that the solution impacted between 5.21% and 5.29% of the total Alexa traffic during the six-week evaluation period where query rewriting was applied. In the same period, Figure 10(b) shows an average defect reduction rate of 18%. These results confirmed the effectiveness of the proposed self-learning framework in improving the Alexa user experience and reducing defects.

Table I presents a collection of query rewrite examples, illustrating various types of defects being automatically resolved by Alexa's self-learning framework. Rows 1 to 4 exemplify instances where ASR errors are corrected through query rewriting. Examples 1 and 2 entail syntactically correct ASR outputs, demanding resolution through contextual and personal context. The third and fourth examples pose greater challenges, each containing two recognition errors that require correction. In Examples 5 and 6, the original utterances are reformatted to ensure accurate parsing by NLU. Rows 7 and 8 demonstrate personalized query rewriting, wherein artists' names are added to disambiguate song entities. This capability is made possible by the self-learning framework, which can effectively infer customer preferences. Finally, the last two examples exhibit Alexa's exceptional ability to rewrite queries solely based on the user's intent, a task that cannot be accomplished through phoneme or word similarity matching methods.

G. Limitation

The self-learning framework empowers conversational AI to continuously learn on its own once it is deployed in the real world. Despite its numerous advantages, the framework has certain limitations. Firstly, it does not tackle the cold start problem, as the AI system needs to be developed and deployed before it can start learning. Data annotation remains a critical step in constructing initial models for various components such as ASR, NLU, and TTS, although smaller datasets suffice. For example, the development of high-quality ASR models typically requires tens of thousands of hours of speech data [62], [63]. Similarly, NLU often demands hundreds of queries per intent [14], [64], and TTS systems often require hundreds of hours of speech data for their development [65]. The self-learning framework excels in continuous learning from real-world interactions, requiring only minimal supervision and annotation (*e.g.,* 20K turns for Alexa) to monitor the feedback interpreter's performance. Secondly, the effectiveness of the framework depends on the volume of interactions between the users and the system. Systems derive greater benefits and improve more rapidly with increased usage.

IV. RELATED WORK

In previous sections, a viable self-learning framework for large-scale conversational AI systems was established, and the process of implementing the proposed framework was described. This section describes related work in other conversational AI systems. Moreover, it includes a discussion on how popular ML paradigms have fallen short in addressing the challenges outlined in Section I. Finally, a review on self-learning related efforts in other AI systems, *i.e.,* web search engines, and recommendation systems, is presented.

A. Other Conversational AI Systems

To the best of our knowledge, the framework proposed in this article is the first to apply self-learning to improve a conversational AI system systematically. However, the following works also involve one or more of the self-learning elements described in Section II-A.

At Microsoft Cortana, fine-grained user signals were harnessed to build a feedback interpreter to predict user satisfaction at the session level. These signals include explicit feedback, responses such as *"yes, thanks"* [66], implicit feedback derived from mobile device interactions such as touches and swipes [67], and acoustic signals [68], *e.g.,* a reduction in utterance speed.

In Yahoo Captain (a short-message-based family assistant), *user engagement status* such as fulfillment, reformulation, and abandonment is predicted at the turn level to estimate the success or failure of sessions [69].

At Apple Siri, several learning mechanisms apply user feedback to improve their experiences. Nguyen *et al.* [70] employed the rewriting mechanism to recover user queries with repeats, which is a specific form of rephrasing. For example, *"Call Uncle of R . . . No I said Uncle LeVar"* could be rewritten as *"Call Uncle LeVar."* Siri also incorporated an attribution mechanism, where users' positive (*e.g.,* listening to a song for over 30 seconds) or negative (aborting the song and switching/re-searching) responses provide implicit feedback that can indicate the accuracy of song searches. This feedback leads to the generation of synthetic labels for named entity recognition and typing components [71].

B. Machine Learning Paradigms

In addressing the challenges outlined in Section I, academia has explored a diverse range of *ML paradigms*, although not all challenges have been fully tackled. For example, *multi-task learning* can be applied to allow components with limited annotations (*e.g.,* the ASR, DM, and response generation components) to benefit from abundant annotations on other tasks (*e.g.,* NLU [72]) or even unlabeled data [73], [74]. The latter approach also falls under the umbrella of *semi-supervised learning*. *Weakly supervised learning* enables model training with relaxed annotation requirements, such as utilizing answer-passage annotations instead of answer-span annotations for conversational question answering [75]. *Self-supervised learning* designs annotation-free data recovery tasks on unlabeled conversations to pretrain models for target tasks, *e.g.,* pretraining via inconsistent order detection for end-to-end response generation [76]. These ML paradigms can partially address Challenge 1 by creating unlabeled data or annotations for other tasks. However, they typically require model retraining, making them less adaptable for dealing with Challenge 5. Moreover, when leveraging annotations from other tasks, they also fall short in addressing Challenges 2 and 3.

In conversational AI, *reinforcement learning* has been applied in dialog management, where a user simulator is initially trained, and then the system is further optimized in an end-to-end manner via a simulator [77]. *Federated learning* techniques eliminate the need to centralize user data when training models for NLU [78], text classification [79], and broader natural language processing (NLP) tasks [80]. Both techniques address Challenge 3 to some extent by restricting the access to sensitive user data to certain stages and devices. *Online learning* [81] and *meta-learning* [82]–[84] tackle Challenge 5 by avoiding the need to retrain ML models from scratch, thus shortening system-improvement cycles. However, compared with the self-learning framework proposed in this article, where manual annotations are replaced with user feedback, none of these ML paradigms aim to eliminate the system's dependency on manual annotations for continuous learning, and therefore still face Challenges 1 and 2.

It is important to note that the proposed self-learning framework is not tied to any particular ML method, and its components can leverage all ML advancements. For example, the feedback interpreter can be implemented using various ML model architectures.

C. Self-learning in Other Systems

While this study represents the first attempt to formalize self-learning as a framework for conversational AI systems, similar concepts have emerged in various other domains. In the following, an attempt is made to align them with the framework components, namely the feedback interpreter, feedback collection, and learning mechanisms, to foster a more comprehensive understanding of this self-learning framework.

1) Search Engines: When users query a search engine, interact with the search engine result pages (SERPs), and reformulate their query, search engines record the user feedback in query logs. Research has been conducted regarding interpreting rich user feedback at different levels of granularity. Similar to conversational AI systems, search engine users often reformulate their queries within short time intervals, creating *search sessions*. At the coarse-grained level, researchers have leveraged query reformulation behavior to build evaluators that predict user satisfaction [85], [86]. Rewriting [87] is implemented as a learning mechanism to intervene in problematic user queries (*e.g.,* correcting typos) before they are executed. At a finer-grained level, the way a user interacts with SERPs and clicks on links in consecutive queries provides a wealth of implicit user feedback [88] in the form of *document preference pairs*. These feedback are then leveraged via the attribution learning mechanism [89]. This mechanism identifies cases where the ranking model may be at fault and should consider these interactions as valuable training data for future improvements. In this line of research toward a better feedback interpreter, dwell time, which measures how long a user spends reading a page after clicking on its link, has been employed as the most fine-grained feedback available in search logs [90].

With feedback in search logs being exhausted, there has been a research trend of accumulating new types of feedback with higher "resolutions" on users' interactions with SERPs. Possible types of such feedback include touches on mobile devices [91], mouse behaviors on desktops/laptops [92], eye tracking histories [93], and even electrodermal signals [94]. With the best trade-off between resolution and availability, mouse behaviors on a SERP page [92], [95], including cursor trajectories, scrolling speeds, and cursor hovers, or even on the landing pages [96], are most commonly employed for search session evaluation. Following the attribution-learning mechanism, trajectory motifs, *i.e.,* frequent sub-sequences of cursor positions, are leveraged to improve search ranking [97]. Unlike conversational AI systems, web search engines typically do not engage in direct conversations with users to solicit feedback. It is also uncommon for users to share unsolicited feedback. Therefore, a dedicated feedback interpreter or collector is not required for web search engines.

2) Recommendation Systems: In the recommendation system literature, user feedback is also categorized as twofold: explicit (*e.g.,* user ratings) and implicit (*e.g.,* clicking history) [98]. From the self-learning perspective, for more informative feedback, it is essential to consider recommendation systems as a component residing in a more extensive system rather than evaluating and improving it in isolation. Such a holistic view will enable richer user feedback (*e.g.,* thumbs-up/down, ratings, immersion time [99], [100], product purchases, location visits, and user reviews [101]) for better experience evaluation and relieve the dependency on user clicks, which have been proven not to necessarily indicate user satisfaction [102].

Regarding learning mechanism, feedback such as ratings have mostly been utilized in model retraining. Meanwhile, the latest efforts in this problem space have been put into explainable recommendations [103] to associate user experiences with product attributes [104], location aspects [105], *etc.*, which fall more into the learning mechanism of attribution. Finally, it should be noted that guardrail and consulting mechanisms are also common in recommendation systems. For example, strongly negative feedback regarding user reviews or reports can trigger manual reviews or bans to prevent harmful content from spreading. Unlike conversational AI systems, recommendation systems provide proactive experiences that are not initiated by users. Implicit feedback such as rephrasing or explicit feedback are not available to evaluate the experience.

V. Conclusion and Looking Forward

In this article, the challenges that industry conversational AI systems face due to their reliance on manual annotations were highlighted. This dependency hinders the ability to achieve continuous, consistent, and prompt improvements. These challenges emerge from an interplay of manual annotations, modular architecture, and high user expectations for large-scale conversational AI systems. Inspired by how humans learn to tackle with complex tasks through experiences, a self-learning framework has been proposed in this article. This framework characterizes elements and patterns that conversational AI systems can adopt to leverage user feedback through various mechanisms. The article also includes a description of how self-learning can be systematically applied to Alexa, showcasing its positive impact on user experiences. Finally, related works in other systems falling within the self-learning framework were discussed.

Although a self-learning solution for Alexa has been successfully implemented, it remains in its early stages. Therefore, it is anticipated that this framework can be expanded to cover more components and scenarios in future work.

- *Extending Defect Attribution.* ASR defect attribution was introduced as an example of the defect attribution mechanism. It is intended to expand this approach to other components through unsolicited feedback. For example, when users make statements such as *"Alexa, I'm not talking to you," "Alexa, I didn't say your name," etc.*, the feedback from previous turns can be attributed to errors in the wake word detection module. Additionally, feedback such as *"Alexa, I'm not Eric," "My name is Dave," etc.*, can be employed to enhance the *speaker identification* module. This improvement helps provide personalized experiences for different users within the same household.
- *Generalizing Solicited Feedback.* Another avenue of future work is exploring how to generalize the consultation mechanism to users. Currently, explicit feedback on query rewrites are solicited through *"Did you mean ...?"* questions. However, expanding this mechanism beyond binary questions holds promise. For example, when the system is uncertain about whether the request *"Play Fearless"* issued to a smart TV remote refers to a song or a movie, the system can clarify the two choices with the user.
- *Leveraging Multimodal Feedback.* At present, the proposed framework's primary means of receiving feedback is through speech and text input. However, there are additional multi-modal channels through which the self-learning framework could be enhanced. One such approach is to collect comprehensive feedback from users' gestures, screen touches, remote control clicks, *etc.*
- *Leveraging Large Language Models.* Most components within the self-learning framework proposed in this article have been built using transformer-based models. Given the promising advancements in large language models (LLMs), there is potential to leverage these LLMs to further enhance conversational AI systems. Self-learning is poised to play a pivotal role in the continuous improvement of these systems, relying on user feedback instead of human annotations.

VI. Acknowledgments

The authors would like to extend their sincere gratitude to their colleagues, including Jacky Yi, Eunah Cho, Jie Hao, Pragaash Ponnusamy, Meiguo Wang, Kun Zhou, Bin Guo, Zongkai Sun, Zhengyang Zhao, Xiyao Ma, Cynthia Lu, Sihong Liu, Chengyuan Ma, Xing Fan, Sawyer Shen, Yu Zhang, Yanbin Lu, and others, for their invaluable contributions to the development of the self-learning framework for Alexa. Special thanks are also extended to Spyros Matsoukas, Nikko Strom, Prem Natarajan, and Gokhan Tur for their insightful discussions and guidance, which have significantly influenced the design and implementation of the framework. The authors are also grateful to Gustavo Aguilar, Jialong Han, and Karen Stabile for their assistance with editing the article, creating figures, and proofreading the article.

References

[1] R. Sarikaya, "The technology behind personal digital assistants: An overview of the system architecture and key components," *IEEE Signal Processing Magazine*, vol. 34, no. 1, pp. 67–81, 2017.
[2] A. Acharya *et al.*, "Alexa conversations: An extensible data-driven approach for building task-oriented dialogue systems," in *NAACL 2021*, 2021, pp. 125–132.
[3] J. Gao, M. Galley, and L. Li, "Neural approaches to conversational ai," in *Foundations and Trends in Information Retrieval*, 2019, pp. 127–298.
[4] M. Mctear and O. Seminck, "Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots by Michael McTear," *Computational Linguistics*, vol. 13, pp. 1–4, 2022.
[5] J. Piaget, *The psychology of intelligence*. Routledge, 2005.
[6] D. Kolb, *Experiential Learning: Experience as the Source of Learning and Development*. Pearson Education Limited, 2015.
[7] M. S. Knowles, "The modern practice of adult education: from pedagogy to andragogy," *New York: Cambridge*, 1980.
[8] L. H. Lewis and C. J. Williams, "Experiential learning: Past and present," *New directions for adult and continuing education*, vol. 1994, no. 62, pp. 5–16, 1994.
[9] D. R. Olson and J. S. Bruner, "Learning through experience and learning through media," *Teachers College Record*, vol. 75, no. 5, pp. 125–150, 1974.

[10] R. Sarikaya, "Intelligent conversational agents for ambient computing," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022.

[11] Y.-B. Kim, D. Kim, J.-K. Kim, and R. Sarikaya, "A scalable neural shortlisting-reranking approach for large-scale domain classification in natural language understanding," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*. Association for Computational Linguistics, Jun. 2018, pp. 16–24.

[12] Y.-B. Kim, D. Kim, A. Kumar, and R. Sarikaya, "Efficient large-scale neural domain classification with personalized attention," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, Jul. 2018, pp. 2214–2224.

[13] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 8689–8696.

[14] X. Zang, A. Rastogi, S. Sunkara, R. Gupta, J. Zhang, and J. Chen, "MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines," in *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, Jul. 2020, pp. 109–117.

[15] P. Budzianowski *et al.*, "MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018, pp. 5016–5026.

[16] J. D. Williams, M. Henderson, A. Raux, B. Thomson, A. Black, and D. Ramachandran, "The dialog state tracking challenge series," *AI Magazine*, vol. 35, no. 4, pp. 121–124, 2014.

[17] E. Hovy and J. Lavid, "Towards a 'science'of corpus annotation: a new methodological challenge for corpus linguistics," *International journal of translation*, vol. 22, no. 1, pp. 13–36, 2010.

[18] T. Paek and R. Pieraccini, "Automating spoken dialogue management design using machine learning: An industry perspective," *Speech communication*, vol. 50, no. 8-9, pp. 716–729, 2008.

[19] B. Shmueli, J. Fell, S. Ray, and L.-W. Ku, "Beyond fair pay: Ethical implications of NLP crowdsourcing," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Jun. 2021, pp. 3758–3769.

[20] E. L. Tonkin, "Keynote: Privacy and security in real-world data annotation," in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2020, pp. 1–1.

[21] G. Kazimzade and M. Miceli, "Biased priorities, biased outcomes: three recommendations for ethics-oriented data annotation practices," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020, pp. 71–71.

[22] A. B. Saka, L. O. Oyedele, L. A. Akanbi, S. A. Ganiyu, D. W. Chan, and S. A. Bello, "Conversational artificial intelligence in the aec industry: A review of present status, challenges and opportunities," *Advanced Engineering Informatics*, vol. 55, 2023.

[23] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," *Advances in neural information processing systems*, vol. 31, 2018.

[24] Y. Ouali, C. Hudelot, and M. Tami, "An overview of deep semi-supervised learning," *arXiv preprint arXiv:2006.05278*, 2020.

[25] L. Chen, F. Garcia, V. Kumar, H. Xie, and J. Lu, "Industry scale semi-supervised learning for natural language understanding," in *NAACL 2021*.

[26] K. Bhattacharjee *et al.*, "To bert or not to bert: Comparing task-specific and task-agnostic semi-supervised approaches for sequence tagging," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, p. 7927–7934.

[27] Y. Zhang, Z. Ou, M. Hu, and J. Feng, "A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Nov. 2020, pp. 9207–9219.

[28] X. Huang, J. Qi, Y. Sun, and R. Zhang, "Semi-supervised dialogue policy learning via stochastic reward estimation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Jul. 2020, pp. 660–670.

[29] G. A. Hasselt Hado van and S. David, "Deep reinforcement learning with double q-learning," in *Proceedings of the Annual AAAI Conference on Artificial Intelligence (AAAI'16)*, 2016, p. 2094–2100.

[30] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937.

[31] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 449–458.

[32] C.-V. Pal and F. Leon, "Brief survey of model-based reinforcement learning techniques," in *2020 24th International Conference on System Theory, Control and Computing (ICSTCC)*, 2020, pp. 92–97.

[33] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *J. Netw. Comput. Appl.*, vol. 116, pp. 1–8, 2018.

[34] J. Ding, E. Tramel, A. K. Sahu, S. Wu, S. Avestimehr, and T. Zhang, "Federated learning challenges and opportunities: An outlook," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 8752–8756.

[35] C. Wang *et al.*, "A secure and efficient federated learning framework for NLP," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021, pp. 7676–7682.

[36] K. Wei *et al.*, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[37] R. Khaziev, U. Shahid, T. Röding, R. Chada, E. Kapanci, and P. Natarajan, "FPI: Failure point isolation in large-scale conversational assistants," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*. Hybrid: Seattle, Washington + Online: Association for Computational Linguistics, Jul. 2022, pp. 141–148.

[38] P. Ponnusamy, A. R. Ghias, C. Guo, and R. Sarikaya, "Feedback-based self-learning in large-scale conversational AI agents," in *The Thirty-Second Annual Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 2020, pp. 13 180–13 187.

[39] S. Gupta *et al.*, "Robertaiq: An efficient framework for automatic interaction quality estimation of dialogue systems," in *2nd International Workshop on Data-Efficient Machine Learning (DeMaL)*, 2021.

[40] M. Wang *et al.*, "Joint goal segmentation and goal success prediction on multi-domain conversations," in *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, 2022, pp. 505–509.

[41] J. Hao *et al.*, "CGF: Constrained generation framework for query rewriting in conversational AI," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, 2022, pp. 475–483.

[42] Y. Zhou *et al.*, "Unified contextual query rewriting," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, Toronto, Canada, Jul. 2023, pp. 608–615.

[43] Y. Liu *et al.*, "Roberta: A robustly optimized bert pretraining approach," 2019.

[44] S. Yuan, S. Gupta, X. Fan, D. Liu, Y. Liu, and C. Guo, "Graph enhanced query rewriting for spoken language understanding system," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 7997–8001.

[45] X. Fan, E. Cho, X. Huang, and C. Guo, "Search based self-learning query rewrite system in conversational ai," in *2nd International Workshop on Data-Efficient Machine Learning (De-MaL)*, 2021.

[46] E. Cho *et al.*, "Personalized search-based query rewrite system for conversational ai," in *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, 2021, pp. 179–188.

[47] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, p. 2333–2338.

[48] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A convolutional latent semantic model for web search," in *Technical Report MSR-TR-2014-55, Microsoft Research*, 2014.

[49] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Transactions on Big Data*, pp. 535–547, 2021.

[50] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations*, 2015.

[51] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2014, p. 3104–3112.

[52] M. Lewis *et al.*, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.

[53] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.

[54] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *Advances in neural information processing systems*, 2011, pp. 2249–2257.

[55] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples." Oxford University Press, 1933, pp. 285–294.

[56] T. Graepel, J. Q. n. Candela, T. Borchert, and R. Herbrich, "Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, 2010, pp. 13–20.

[57] D. N. Hill, H. Nassif, Y. Liu, A. Iyer, and S. Vishwanathan, "An efficient bandit algorithm for realtime multivariate optimization," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1813–1821.

[58] J.-K. Kim, G. Wang, S. Lee, and Y.-B. Kim, "Deciding whether to ask clarifying questions in large-scale spoken language understanding," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021, pp. 869–876.

[59] S. Park *et al.*, "A scalable framework for learning from implicit user feedback to improve natural language understanding in large-scale conversational ai systems," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 6054–6063.

[60] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2020.

[61] R. Kohavi and S. Thomke, "The surprising power of online experiments," in *Harvard Business Review*, 2017, p. 74–82.

[62] J. Kahn *et al.*, "LIBRI-LIGHT: a benchmark for asr with limited or no supervision," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2020, pp. 7669–7673.

[63] T. Javed *et al.*, "Towards building asr systems for the next billion users," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, 2022, pp. 10 813–10 821.

[64] S. Louvan and B. Magnini, "Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey," in *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, 2020, pp. 480–496.

[65] H. Zen *et al.*, "Libritts: A corpus derived from librispeech for text-to-speech," 09 2019, pp. 1526–1530.

[66] S. H. Hashemi, K. Williams, A. El Kholy, I. Zitouni, and P. A. Crook, "Measuring user satisfaction on smart speaker intelligent assistants using intent sensitive query embeddings," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1183–1192.

[67] J. Kiseleva, K. Williams, A. Hassan Awadallah, A. C. Crook, I. Zitouni, and T. Anastasakos, "Predicting user satisfaction with intelligent assistants," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 45–54.

[68] J. Jiang *et al.*, "Automatic online evaluation of intelligent assistants," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 506–516.

[69] R. Meng, Z. Yue, and A. Glass, "Predicting user engagement status for online evaluation of intelligent assistants," in *European Conference on Information Retrieval*. Springer, 2021, pp. 433–450.

[70] H. L. Nguyen, V. Renkens, J. Pelemans, S. P. Potharaju, A. K. Nalamalapu, and M. Akbacak, "User-Initiated Repetition-Based Recovery in Multi-Utterance Dialogue Systems," in *Proc. Interspeech 2021*, 2021, pp. 226–230.

[71] D. Muralidharan *et al.*, "Leveraging user engagement signals for entity labeling in a virtual assistant," in *NeurIPS 2018 Workshop on Conversational AI*, 2018.

[72] A. Rastogi, R. Gupta, and D. Hakkani-Tur, "Multi-task learning for joint language understanding and dialogue state tracking," in *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, 2018, pp. 376–384.

[73] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," *Proc. Interspeech 2018*, pp. 2–6, 2018.

[74] C. Zhu, M. Zeng, and X. Huang, "Multi-task learning for natural language generation in task-oriented dialogue," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1261–1266.

[75] C. Qu, L. Yang, C. Chen, W. B. Croft, K. Krishna, and M. Iyyer, "Weakly-supervised open-retrieval conversational question answering," in *European Conference on Information Retrieval*. Springer, 2021, pp. 529–543.

[76] J. Wu, X. Wang, and W. Y. Wang, "Self-supervised dialogue learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3857–3867.

[77] X. Li, Y.-N. Chen, L. Li, J. Gao, and A. Celikyilmaz, "End-to-end task-completion neural dialogue systems," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2017, pp. 733–743.

[78] Z. Huang, F. Liu, and Y. Zou, "Federated learning for spoken language understanding," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 3467–3478.

[79] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: A comprehensive review," *ACM Comput. Surv.*, vol. 54, no. 3, 2021.

[80] B. Y. Lin *et al.*, "FedNLP: Benchmarking federated learning methods for natural language processing tasks," in *Findings of the Association for Computational Linguistics: NAACL 2022*. Association for Computational Linguistics, 2022, pp. 157–175.

[81] P.-H. Su *et al.*, "On-line active reward learning for policy optimisation in spoken dialogue systems," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 2431–2441.

[82] S. Lin *et al.*, "Graph-evolving meta-learning for low-resource medical dialogue generation," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*. AAAI Press, 2021, pp. 13 362–13 370.

[83] K. Qian and Z. Yu, "Domain adaptive dialog generation via meta learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2639–2649.

[84] J. Zeng, Y. Yin, Y. Liu, Y. Ge, and J. Su, "Domain adaptive meta-learning for dialogue state tracking," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2493–2501, 2021.

[85] A. Hassan, X. Shi, N. Craswell, and B. Ramsey, "Beyond clicks: query reformulation as a predictor of search satisfaction," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 2019–2028.

[86] J. Chen *et al.*, "Incorporating query reformulating behavior into web search evaluation," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 171–180.

[87] X. Wang and C. Zhai, "Mining term association patterns from search logs for effective query reformulation," in *Proceedings of the 17th ACM conference on Information and knowledge management*, 2008, pp. 479–488.

[88] T. Joachims and F. Radlinski, "Search engines that learn from implicit feedback," *Computer*, vol. 40, no. 8, pp. 34–40, 2007.

[89] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 133–142.

[90] Y. Kim, A. Hassan, R. W. White, and I. Zitouni, "Modeling dwell time to predict click-level satisfaction," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 193–202.

[91] Y. Zheng, J. Mao, Y. Liu, M. Sanderson, M. Zhang, and S. Ma, "Investigating examination behavior in mobile search," in *Proceedings of the 13th international conference on web search and data mining*, 2020, pp. 771–779.

[92] Q. Guo, D. Lagun, and E. Agichtein, "Predicting web search success with fine-grained interaction data," in *Proceedings of the 21st ACM*

international conference on Information and knowledge management, 2012, pp. 2050–2054.

[93] L. A. Granka, T. Joachims, and G. Gay, "Eye-tracking analysis of user behavior in www search," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, 2004, pp. 478–479.

[94] Y. Wu, Y. Liu, Y.-H. R. Tsai, and S.-T. Yau, "Investigating the role of eye movements and physiological signals in search satisfaction prediction using geometric analysis," *Journal of the Association for Information Science and Technology*, vol. 70, no. 9, pp. 981–999, 2019.

[95] Y. Liu *et al.*, "Different users, different opinions: Predicting search satisfaction with mouse movement information," in *Proceedings of the 38th international acm sigir conference on research and development in information retrieval*, 2015, pp. 493–502.

[96] Q. Guo and E. Agichtein, "Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior," in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 569–578.

[97] D. Lagun, M. Ageev, Q. Guo, and E. Agichtein, "Discovering common motifs in cursor movement data for improving web search," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 183–192.

[98] D. W. Oard, J. Kim *et al.*, "Implicit feedback for recommender systems," in *Proceedings of the AAAI workshop on recommender systems*, vol. 83. AAAI, 1998, pp. 81–83.

[99] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *2008 Eighth IEEE international conference on data mining*. Ieee, 2008, pp. 263–272.

[100] X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan, "Beyond clicks: dwell time for personalization," in *Proceedings of the 8th ACM Conference on Recommender systems*, 2014, pp. 113–120.

[101] R. L. Rosa, D. Z. Rodriguez, and G. Bressan, "Music recommendation system based on user's sentiments extracted from social networks," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 3, pp. 359–367, 2015.

[102] Q. Zhao, F. M. Harper, G. Adomavicius, and J. A. Konstan, "Explicit or implicit feedback? engagement or satisfaction? a field experiment on machine-learning-based recommender systems," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 2018, pp. 1331–1340.

[103] Y. Zhang, X. Chen *et al.*, "Explainable recommendation: A survey and new perspectives," *Foundations and Trends® in Information Retrieval*, vol. 14, no. 1, pp. 1–101, 2020.

[104] Y. Xian *et al.*, "Ex3: Explainable attribute-aware item-set recommendations," in *Fifteenth ACM Conference on Recommender Systems*, 2021, pp. 484–494.

[105] C. Musto, M. de Gemmis, G. Semeraro, and P. Lops, "A multi-criteria recommender system exploiting aspect-based sentiment analysis of users' reviews," in *Proceedings of the eleventh ACM conference on recommender systems*, 2017, pp. 321–325.