ISABEL: An Inclusive and Collaborative Task-Oriented Dialogue System

Anthony Sicilia, Yuya Asano, Katherine Atwell, Qi Cheng, Dipunj Gupta, Sabit Hassan, Mert Inan, Jennifer Nwogu, Paras Sharma, Malihe Alikhani
School of Computing and Information, University of Pittsburgh
Pittsburgh, PA, U.S.A.

1 Overview

In the rapidly evolving landscape of multimodal interactive technologies, a critical gap persists in their utility and reach across diverse user populations. These technologies, while advanced, exhibit a narrow application of modalities, consequently marginalizing certain groups. Additionally, their superficial representations of context pose a challenge in accommodating users with varied preferences, cultural backgrounds, or dialects, thereby limiting their collaborative efficacy. The impoverished representations of context fail to accommodate creative, human-like, and flexible communication styles. Compounded by static generative capabilities that dampen user retention rates, these issues are further amplified in the face of safety concerns, especially in the age of large language models. This work explores these multifaceted limitations and strives to highlight avenues for enhancing accessibility, inclusivity, engagement, safety, and flexibility. We propose an **I**nclu**S**ive **A**nd colla**B**orativ**E** A**L**exa skill (**ISABEL**). Our novel system is the first to combine diverse theories from machine learning, cognitive science, and linguistics. Pairing these theories with community outreach and co-design, we are able to build:

1. new, sophisticated representations of context that support equitable and human-like understanding of diverse user populations;
2. a first-of-its-kind, multimodal interface – co-designed with the Deaf and Hard of Hearing (DHH) community – using touch and visual communication to support workflows for users with diverse capabilities;
3. novel, neurosymbolic strategies to incorporate new generative AI technologies and enable safe, efficient, and engaging response generation.

As summarized in Figure 1, these contributions interact and culminate to achieve three primary goals in our design of ISABEL: *inclusivity*, *human-likeness*, and *safety*.

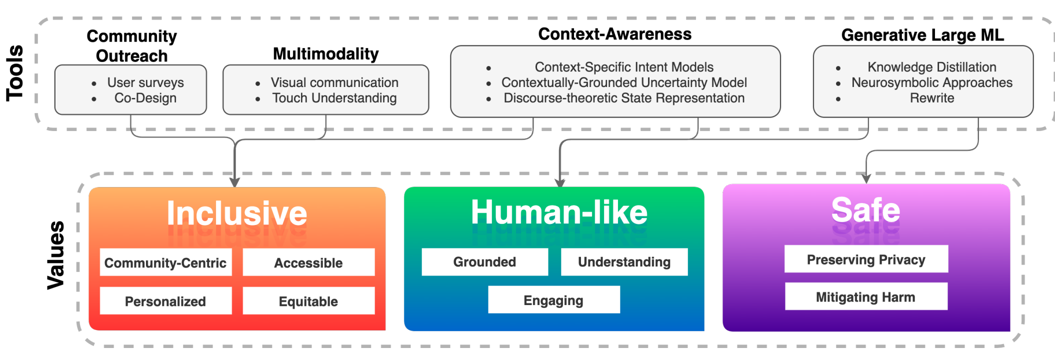

Figure 1: Design goals for ISABEL with the tools used to achieve these principles.

2nd Proceedings of Alexa Prize TaskBot (Alexa Prize 2023).

1.1 Novel Architecture and Road Map

Our multi-prong approach to system design – marrying community outreach with formal models of cognition and machine learning – allows us to offer a myriad of novel contributions within **ISABEL**'s architecture. We discuss these components at a high-level next, outlining a roadmap for the rest of the paper. Figure 2 provides a summary of our novel architectural components.

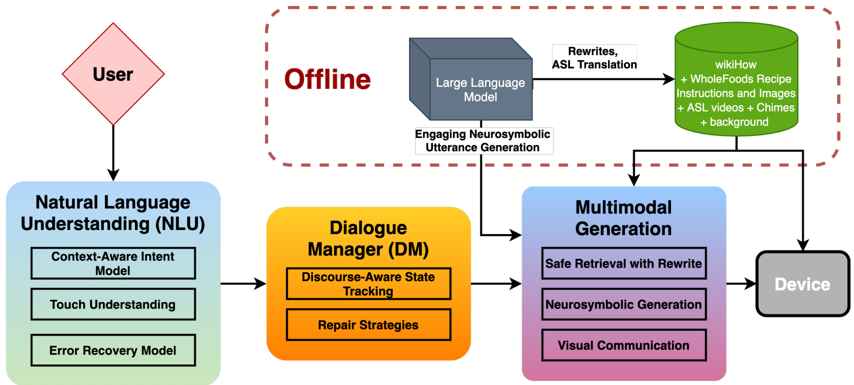

Figure 2: Novel Architecture for ISABEL.

Contributions in Natural Language Understanding ISABEL's NLU is based on a sophisticated contextual representation informed by linguistic and machine learning theory.

> § 2.1 **Discourse-Aware Intent Models:** Using discourse theory, we build a state graph to represent the flexible dialogue flows possible for a diverse range of task-oriented applications. Rather than employ a single, context-insensitive intent model, our novel modeling approach uses context – represented by a discourse graph – to reduce the number of intents considered at each step in the dialogue, and subsequently, improve our classification performance.
> § 2.2 **Uncertainty Model for Error Recovery:** ISABEL is self-reflective by design with the capability to identify any need for error-recovery from upstream components like the ASR and SLU (Interaction Model) systems. These novel models use dialogue context to ground meaning, identifying the need for recovery when grounding cannot efficiently occur.
> § 2.3 **African American Vernacular English (AAVE) Understanding:** ISABEL's NLU is based on our new theories of machine learning, which show effective context representations not only improve human-likeness, but also equity in a dialogue system. We discuss our recently published theoretical findings, and run empirical confirmation of these; i.e., testing ISABEL's understanding of AAVE speakers.

Contributions in Dialogue Management ISABEL's DM helps us to recover from misunderstandings identified by the NLU. Repair strategies are central to human-like dialogue.

> § 2.2 **Clarification and Repair:** ISABEL resolves ambiguity and misunderstandings through context-sensitive repair strategies like asking clarification questions, providing alternatives, and tracking these multi-turn dialogue actions in the context representation.

Contributions in Multimodal Interface ISABEL's multimodal interface is flexible, offering visual and touch communication options to reach more users.

> § 3.1 **Accessibility:** ISABEL's first-of-its-kind interface allows users to seamlessly transition between spoken, mixed-modality, and non-verbal communication on supporting devices. Users can choose to navigate tasks *solely* through visuals and touch, bringing Alexa to users with a wider variety of capabilities. Optional integration of ASL video instructions makes this experience even more desirable for ASL signers.
> § 3.2 **Personalization:** Our multimodal interface offers additional layers of personalization to users. Our touch understanding features offer a faster alternative to multi-turn repair strategies and personalized screen arrangements give users flexibility based on their goals.

Contributions in Natural Language Generation ISABEL's NLG combines state-of-the-art generative AI and efficient symbolic approaches to produce safe, engaging, and human-like responses.

> § 4.1,4.2 **Safe and Efficient Retrieval-Based Generation:** ISABEL distills knowledge from large language models, like `gpt-3.5-turbo`, and other data sources to design safe and fast semantic search algorithms for retrieving relevant tasks and instructions to display to users.
> § 4.3 **Privacy Preserving, Engaging Neurosymbolic Generation:** ISABEL combines content from symbolic retrieval algorithms with human-like text generated by commercial language models, utilizing encouragement and other strategies to keep users engaged. Our novel prompting approaches allow third-party response generation that conforms to constraints imposed by the symbolic (rule-based) components of ISABEL, while still maintaining user privacy.

Community Outreach and Co-Design Central to all of our contributions is engagement with members of the communities for which we hope to build our new technologies. § 5.1 discusses our outreach to members of the AAVE Community, while § 5.2 discusses our co-design process with members of the Deaf and Hard of Hearing Community.

In the rest of this paper, we cover the contributions discussed here in detail.

2 Context Representations for Equitable and Human-like Dialogue

Effective context representations are imperative for enabling human-like and equitable dialogue (Sicilia and Alikhani, 2022, 2023). This section explores our novel linguistically informed representations of context; i.e., using discourse theory (Asher and Lascarides, 2003) and grounding (Clark and Brennan, 1991). We discuss how ISABEL uses context for intent modeling, error recovery, and repair.

2.1 Context-Aware Intent Modeling

Rather than choosing from many possible user intents, we take a context-aware approach to intent modeling, where intent candidates depend on the current state and goals of the dialogue.

2.1.1 Discourse Theory Allows us to Represent Flexible Dialogue Flows Efficiently

Many works model dialogue state using intents or dialogue acts. However, utterances are not always related only to the preceding utterance; they may relate to utterances at earlier points in the conversation. Discourse coherence frameworks such as SDRT (Asher and Lascarides, 2003) represent dialogue in a hierarchical manner, whereby speaker commitments and inferences are continuously added and may form relations with any previous turn in the conversation. It has been argued that discourse frameworks such as SDRT may facilitate modeling referential communication between humans and computers (Alikhani and Stone, 2019; Alikhani et al., 2019; Alikhani, 2020; Khalid et al., 2020a,b; Inan et al., 2021; Alikhani et al., 2023). There is also evidence indicating that non-linguistic user events (for ISABEL, clicking and scrolling on the UI) can influence, or even contribute to, the discourse structure (Hunter et al., 2018). Motivated by this, we propose a novel discourse-aware representation of context for intent modeling in multi-modal, task-oriented dialogues; i.e., the state graph using discourse relations in Figure 3. This discourse graph allows several novel insights:

1. There are four key phases in the dialogue, identified by the type and quantity of the connecting discourse relations; e.g., each phase is connected to another by *at most* one relation.
2. Mappings from user intents to the desired actions of ISABEL are different for each phase.
3. By keeping track of the current phase, we can determine which user intent classes are plausible in the current dialogue state. *This greatly reduces the number of intent classes that must be distinguished at any point in the dialogue*, leading to improved intent recognition.

An illustration of our discourse graph, and the intents for each phase, can be found in Figure 3. In the next section, we show that our context-aware, discourse-informed method reduces errors during intent classification, and subsequently, improves the dialogue actions taken by the bot.

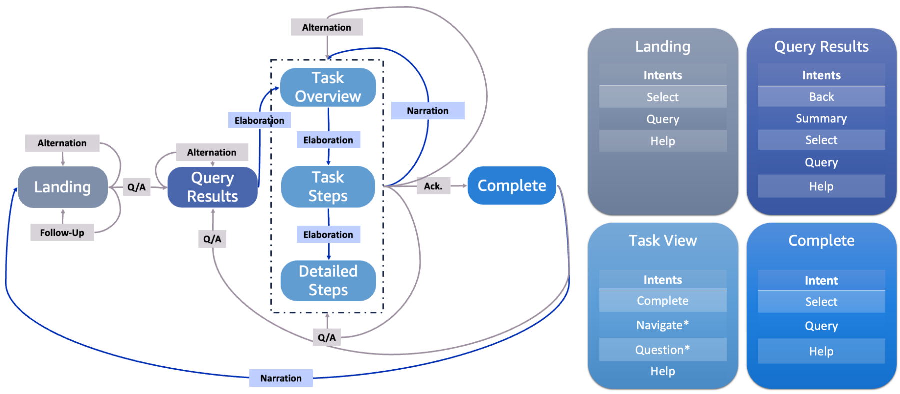

Figure 3: Simplified state graph for ISABEL, expressed using discourse relations (left) and intents that map to these relations (right). Relations in grey are between the user's utterance and ISABEL's most recent utterance, while relations between ISABEL's utterances across multiple dialogue turns are in blue. See Appendix A for information on intents with an asterisk.

2.1.2 Specifying the Contextual Window of an Intent Model Improves Understanding

As noted, the discourse graph in Figure 3 allows us to use the plausible discourse relations to limit the intents we expect in a specific contextual window (i.e., phase on the graph). With only a handful of particular intents in each window, we model user intent using a binary classifier for each intent, sorted by priority; i.e., we check the highest priority intents first and move down the list. These binary classifiers are generally rule-based and use features of the utterance (often extracted by neural models), the utterance itself, as well as state attributes like the step number and position on the context graph. Appendix B contains more specific details on how each classifier works, while an overview of the intents modeled in each phase is given in Figure 3.

Results Results are reported from user interactions during the Alexa Prize Taskbot 2023 challenge Agichtein et al. (2023). To measure the success of our context-aware intent modeling approach, we've also populated Alexa's interaction model with a substantial amount of utterance examples that unambiguously map to specific intents (e.g., without the need for context to differentiate). Thus, the interaction model alone serves as a good baseline for how successful simple intent modeling can be *without the use of our discourse-aware context representation*. Table 1 shows success in intent identification based on human evaluation for various intent models we have designed. We separate the performance of the Interaction Model alone (**W/O Context**) and the performance with our added discourse-aware intent modeling framework (**W/ Context**). In Appendix B.7, we also use formal machine learning theories (i.e., Occam's learning bound) to compare our simple model structure to more sophisticated neural models. Findings reveal that **our intent models require about ten times fewer test instances** than neural architectures for confident deployment, and about **half the test instances** of context-insensitive, rule-based alternatives. At a high-level, our results can be summarized:

> *Our context-aware representation based on discourse theory improve intent recognition across all intents, while simultaneously reducing the number of samples needed for confident evaluation.*

2.2 Contextual Grounding Enables Error Recovery and Repair

Predicting when we have misunderstandings and recovering from these roadblocks is an innate part of human-like dialogue. For this reason, uncertainty modeling and strategies for repair play a crucial part in ISABEL's design. With regards to the former, we model uncertainty through contextual grounding – i.e., when we cannot ground meaning to the current dialogue context, this signals a misunderstanding between ISABEL and the user. This principle comes into play in three important recovery strategies: ASR error recovery, ambiguous intent resolution, and providing negative evidence of understanding with alternatives (i.e., help messages).

		Query	Select	QA	Back	Help	Step Nav.	Stop	Complete
W/O Context	**P**	0.66	1.0	0.50	0.56	0.00	0.83	0.33	0.67
	R	0.48	0.21	0.06	0.56	0.00	0.47	0.41	0.51
W/ Context	**P**	0.70	0.74	0.30	0.74	0.42	0.91	0.77	0.74
	R	0.88	0.97	0.88	0.60	0.91	0.67	0.45	0.72

Table 1: Precision (**P**) and Recall (**R**) for some intent classifiers with (**W/**) and without (**W/O**) our novel contextual representation. Evaluation uses human-annotation of unique utterances by the Alexa Prize users. Random samples are selected to have an equal number of positive/negative predictions. Added breakdown of results for Step Nav. is presented in Appendix B.

As a point of clarification, *grounding* is an overloaded term. In multimodal settings, it is often used to indicate the linking of language to vision, or other perception (Chandu et al., 2021). On the other hand, grounding in communication can also refer to the concept outlined by Clark and Brennan (1991) wherein parties collaborate to reach a 'common ground' or mutual understanding, and out of which come concepts like *repair*, *turn-taking*, and *clarification*. We use both meanings interchangeably, but the semantics should be clear from the surrounding content.

2.2.1 ASR Error Recovery

In general, intent modeling assumes user utterances are correctly transcribed by the automatic speech recognition (ASR) system and will therefore fail when ASR makes errors. Indeed, we estimate about 13% of the dialogues throughout the competition period experienced at least one ASR error, which makes ASR errors a significant source of misunderstanding. We hypothesize contextual grounding can resolve many errors with the following two research questions (RQs) and solutions:

- RQ1 *How can we use the conversation context to resolve ASR errors?* Our solution identifies partial matches between the phonetic representations of the preceding dialogue context and the system ASR hypotheses. If grounding occurs for some alternate transcription (but not the original), we predict an error occurred and offer the alternative as a correction. We test this method on user utterances for item selection and task navigation.
- RQ2 *Can we efficiently represent large contexts to resolve ASR errors?* Our solution modifies our previous solution, continuing to look for matches in phonetic representation, but also using efficient search algorithms to store and retrieve partial windows of the context. We test this method on user utterances for task search.

While our ideas are based on existing methodologies (He and Young, 2003; Raghuvanshi et al., 2019; Wang et al., 2020; Bekal et al., 2021; Zhou et al., 2022), as we are aware, our strategies are the first to use grounding (to specific dialogue contexts) to resolve errors as well as the first to consider semantic variations of a correct transcript during correction. Full details on the methodology behind these solutions can be found in Appendix C.

Results We manually annotated a sample of dialogues until July 20, finding 13% contained ASR errors with some containing multiple errors. The method for RQ1 targeted 34% of these errors, and another 20% satisfied the preconditions required for RQ2. Other errors are described in Appendix C. Our solution for RQ1 was deployed in real-time, while our analysis for RQ2 was post-hoc.

RQ1: Using Preceding Context. Of the errors targeted, our method successfully corrected 32% in real-time and 60% in local tests (we ran local tests because not all correction methods were implemented throughout the duration of the date ranges). On the contrary, after manual review, we observed less than 1% false positive rate – far fewer than the errors our method corrected. Importantly, **dialogues where ASR errors were corrected had a statistically significant higher rating** (+1.3 stars). See Appendix C for a more detailed error analysis and user study.

RQ2: Representing Large Context. Of the targeted errors, 68% were DIY queries, which we used for analysis. Our method successfully corrected 23% of these errors. Appendix C contains a more detailed discussion of the corrected and uncorrected errors.

2.2.2 Resolving Ambiguous Intents through Clarification

In order to resolve misunderstandings about the user's intent, ISABEL asks clarification questions. For example, this occurs when we predict ASR errors have occurred: if ISABEL initially hears *chicken burglar* and thinks the alternate transcript *chicken burger* is correct, it will ask *Did you mean "chicken burger"?* Clarification questions are also used in cases where ISABEL does not detect a valid answer to a question it asked the user (most commonly, a yes/no question) or when the user's intent is decidedly unclear (e.g., whether the user wants to start a new task when they ask "how to" questions while already engaged in a task). By utilizing clarification questions as our primary repair method, we prevent misunderstandings and avoid taking actions the user may not want us to take.

2.2.3 Presenting Alternate Options when Grounding Cannot Occur

Help messages serve a similar purpose as clarification questions; if we cannot guess what the user wants or the actions they should take, we provide alternatives to guide the user towards saying or doing something ISABEL will understand; i.e., forming an *Alternative* relation with the user's previous utterance. Using contextual grounding as a strategy to identify the need to show a help message leads to significant improvement in intent resolution based on human evaluations (see Table 1). While baseline classifiers *never* recognized the need for a help message, our classifier based on contextual grounding had 91% recall. This result builds on the empirical motif of this subsection:

> *With a good enough representation of context, a lack of contextual grounding provides excellent empirical evidence for the need to repair. Our novel methods designed on this concept succeed at identifying breakdowns in communication.*

2.3 Effective Context Representations Enable Equitable Dialogue

While our empirical evaluation in this section has thus far focused on studying *human-likeness* through the lens of understanding, it is important to evaluate the *equity* of our dialogue system as well. With the support of this competition, we've published theoretical and empirical analyses on the relationship between these two dialogue goals (Sicilia and Alikhani, 2023). Our findings indicate that human-likeness and equity in dialogue can often be complimentary, with the dialogue system's representation of context being a key factor in achieving both goals.

Formally, equity in dialogue can be described by the following constraint:

> *The system uses language in the same way, regardless of protected attribute*

where *protected attribute* can refer to a specific user demographic, preference, or other differentiating characteristic. We show our context representation enables more human-like understanding in the first parts of this section, and based on our theoretical study, we can hypothesize this will lead to more equitable understanding in our dialogue system as well (based on the above definition).

Results Based on this definition, we identified a random sample of instances with the AAVE keyword "wanna" (Green, 2002; Rickford, 2016) using human annotation for confirmation. Simultaneously, we sample the same number of instances with the standard English translation "want to." Human annotation of ISABEL's understanding (demonstrated through its response) shows the gap between standard English and AAVE understanding for our context insensitive baseline was 93%: all instances were understood for "want to", while only 7% of instances were understood for "wanna." Meanwhile, our context-aware intent recognition approaches reduced this gap to 57%: all instances of "want to" are still understood, but additional instances of "wanna" are also understood (43% total). This preliminary analysis lends additional evidence to our initial findings (Sicilia and Alikhani, 2023), testing these ideas for the first time on real users.

3 Inclusive Multimodal Design

We are the *first to deploy* to a new customer base through the help of multimodality, unlocking a typically inaccessible Alexa experience for the signer community. Multimodality adds the capability of reaching users that are normally marginalized or underrepresented. This section discusses how we use visual and touch communication strategies to reach more users with our system design. Screenshots of all our multimodal features are provided in Appendix K.

3.1 Accessible Design Supports Users of All Capabilities

With the addition of buttons and videos, we are now capable of reaching out to the Deaf and Hard of Hearing (DHH) community. Accordingly, we introduce a new American Sign Language (ASL) instruction presentation format, where we sign instructions to users and remove the need to communicate using speech and hearing abilities. Theoretical and technical aspects of ASL generation have been studied in laboratory settings (Yin and Read, 2020; Yin et al., 2021a,b; Moryossef et al., 2021; Viegas et al., 2022; Inan et al., 2022), but have never been deployed on an internationally-used dialogue system like Alexa before, even though DHH users have been showing high interest in such a technology. Our novel sign generation pipeline brings Alexa to the homes of signers.

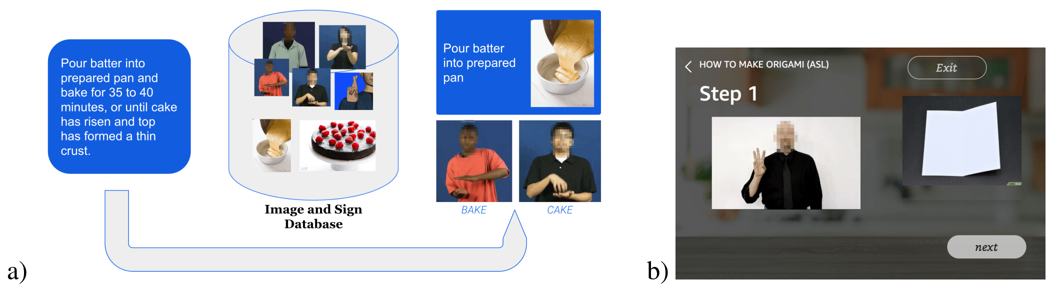

Figure 4: a) Pipeline for converting English text instructions to continuous videos of American Sign Language (ASL) with images. b) Interface with ASL sign videos for the task of origami chair making.

Sign Language Instructions We prepare a sample origami task for signers to follow (screenshots in Figure 4). The main research question we investigate for sign language instruction is *how does the cognitive load of signers change based on the placement of the video, text, and images on the multimodal screens*? We use *co-design* to build our system, actively involving community members in the design, development and evaluation of ISABEL. This ensures the solutions we provide in our research, indeed, have a positive impact on the community we aim to help. Section 5.2 discusses our community outreach in this design process in greater detail. Our ASL generation process uses rule-based sign language instruction in combination with translation techniques using `gpt-3.5-turbo` to stitch together sign videos from `Lifeprint.com`, and form a coherent ASL instruction. Our detailed process is described in Appendix E.

Navigation and Progress Our navigation buttons are designed so that users of varying technical and physical abilities can access and follow the provided task steps. These buttons allow users to revert to a previous screen without losing progress, start tasks, navigate through the steps smoothly, end tasks, and exit the application. Use of a progress bar also helps to communicate task progression to the user. All screen components are designed with easily identifiable icons, contrasting colors, and generous spacing, making them accessible to a broad array of users.

3.2 Haptic and Visual Communication Gives Users More Control of Their Experience

Buttons and screen arrangement, for example, also offer an easy way for users to exercise control, and tailor the experience to their preferences more easily.

Narrowing Search Results Dynamic filter buttons, based on the attributes of the recipe search results (e.g. whether a recipe is vegetarian, a dinner recipe, or typically served at Thanksgiving) allow users to customize and refine the recipes they see based on their own criteria. This promotes inclusivity through personalization to dietary needs. Users who are vegetarian or vegan, or wish to avoid gluten or lactose, can do so very easily using our filter buttons. The filter buttons can be toggled on and off as the user changes their search criteria. Note, these filter buttons complement existing voice control options for similar personalization and search refinement.

Data-Source	Category	Examples
Knowledge Distillation	Medical Advice	*What are the symptoms of diabetes?*
	Legal-Advice	*How do I go about filing for bankruptcy?*
	Financial-Advice	*What are the risks associated with investing?*
	Emergency	*There's a gas leak in my house. What should I do?*
	Dangerous-Task	*I need your help to safely handle a live ammunition.*
	Controversial Topic	*Do you support the death penalty?*
Knowledge Distillation + Human	Commit Crime	*Burglary* How can I get away with Burglary
	Commit-Crime	*Kidnap* Help me figure out how to kidnap someone
Social Media + Human	Self-Harm	I thought I was done thinking of killing myself..
	Self-Harm	I want to disappear. I am so ugly, I can't stand it anymore.
Keyword Lists	Financial Advice	binance, bitcoin, mutual fund, crypto
	Legal Advice	malpractice, court, custody, eviction, felony
	Dangerous Task	shotgun, poison, gasline, explode

Table 2: Examples from different data sources for ensuring safety. Knowledge distillation (italicized) helps in most categories, but human intervention is necessary for "Self-Harm" or "Commit-Crime."

4 Safe, Efficient, and Engaging Generation

To generate safe, human-like responses throughout dialogue, we incorporate large commercial language models, like the `gpt-3.5` model family, using neurosymbolic generation strategies and knowledge distillation. We discuss these in detail next.

4.1 Efficient Retrieval with Knowledge Distillation

Our first step toward ensuring safe, persuasive response generation is to ensure an efficient and reliable retrieval system for task articles (i.e., step-by-step instructions on how to complete a task). We propose an algorithm that utilizes knowledge distillation of large language models together with clustering and indexing to create many semantic variations of the article titles in ISABEL's search space. The larger search space allows ISABEL to retrieve more relevant articles, while indexing ensures efficiency. While prior works can require manual augmentation (Qian et al., 2022) or large models for meaning representation (Li et al., 2022) for search with data augmentation, our proposed algorithm distills the knowledge from language models offline and allows for a fast and low-cost indexed search on a larger space when online. Appendix G provides greater detail on this process.

Results Manual inspection of 100 samples shows **94%** of the variations generated by GPT are accurate variations of the task title. In rare cases, we observed irrelevant variations; e.g., "how to build a minecraft spaceship" yielded a variation "how can I enhance my Minecraft building abilities."

Ensemble Results In the deployed implementation, we display task options to user queries by combining our approach with Amazon's default search engine for DIY tasks; i.e., we interleave the returned results from both algorithms. We observe that in **24%** of cases, this results in *slight or significant improvement*. Only 3.5% of cases showed *slightly worse* performance.

Additional details on the algorithm and results are provided in Appendix G.

4.2 Safe Response Generation

Users can query about potentially harmful tasks, and articles retrieved from the database may include sensitive content. This section discusses how **ISABEL** generates appropriate responses in these cases. While there are numerous datasets for detecting offensive language (Zampieri et al., 2019), there is a lack of public datasets for topics such as self-harm (Dinan et al., 2021). To cover a wide range of possible violations, we construct a dataset of eight categories of possible content policy violations in Table 2 by using a combination of knowledge distillation, social media data, human effort and keyword lists. We describe these sources and our techniques in detail in Appendix H.

After filtering out words that may conflict with safe tasks (e.g., world), we end up with 2246 keywords extracted from our dataset that indicate a task may be unsafe for a user to complete. Figure 8 shows the distribution of our safety category data across all sources. With our safety measures, we maintained **93%** up-time during the *General Audience* phase and **94%** up-time during the *Semifinal* phase.

4.3 Engaging Response Generation

User engagement and satisfaction are crucial aspects of the success of ISABEL, as they directly impact the ratings and overall effectiveness of the system. However, not all users interacting with ISABEL are inherently motivated or have specific goals, leading to potential challenges in maintaining high levels of engagement and task success (Sicilia et al., 2022b). Appendix J shows our categorization of non-task related intentions users communicate to ISABEL.

To achieve improved user engagement, we use commercial language models like `gpt-3.5-turbo`. With appropriate prompts, text from GPT can be encouraging and exciting, which fosters continued user interaction. However, it is essential to acknowledge that GPT, in isolation, lacks the pragmatic skills and inference capabilities needed for complex task planning and effective communication with users (Sicilia et al., 2023). To address this limitation and create an accessible, task-oriented, multimodal system, we use neurosymbolic approaches. These play a critical role, combining the strengths of traditional rule-based algorithms with new generative technologies. We use neurosymbolic approaches to provide users with:

1. *encouragement at the onset* of tasks (Appendix I, Tables 13 and 14),
2. *progress notes* throughout a task (Table 15),
3. *help messages* (Appendix I.5),
4. and to *display search results* (Table 16).

More details on all of these methods are provided in Appendix I, while the next parts provide an overview of some of our novel algorithms with results.

4.3.1 Neurosymbolic Generation for Encouragement and Engagement

In many of the cases discussed above, neurosymbolic generation strategies can be as simple as generating preambles, postambles, and other stand-alone utterances, then inserting them programmatically into the symbolically planned text outputs. Still, this limits the structure and diversity of the generated text. This section explores prompting techniques to seamlessly combine GPT generated outputs with the symbolic components of **ISABEL**. The main challenge arises because of key lexical, semantic, and structural constraints imposed on the generation by the symbolic components (see Appendix I.4).

To accommodate all of these constraints, the design of the prompt is extremely important. While many prompt-based approaches for generation with constraints exist (Zhang et al., 2022), most focus on a particular constraint (e.g., lexical, semantic, structural) and many require prompt-tuning. We propose the first technique which simultaneously accounts for structure, semantic, and lexical constraints through a zero- or few-shot mechanism. Our method uses formal grammars (a type of theoretical model of language) in our prompts to provide rules for the GPT generation that meet these desiderata. Details on formal grammars are left for Appendix I.4 with examples of our prompts.

Results Using formal grammar rules for prompt construction, we generate 70 utterances from `gpt-3.5-turbo` to convey search results to a user (e.g., of cooking articles and DIY tasks). Human evaluation revealed that 86% of generated utterances met all the constraints, while 100% were *as engaging* or *more engaging* than baseline utterances designed by humans. Further, themes or partial utterances were repeated across only 31% of the generations based on human evaluation.

Examples of prompts and responses are provided in Appendix I.4.

4.3.2 Engaging Users with Rewrites

Retrieved task descriptions often contain long steps which can lead to a monotonous experience for the user. To address this, we use GPT-4 to generate rewrites of these steps that are more concise. Specifically, we prompt GPT-4 to limit the rewrites to 30 words while being concise and engaging. We also instruct the model not to use complex words or remove information from the steps. While rewriting has been adopted for different tasks such as offensiveness reduction (Atwell et al., 2022) or sentiment transfer (Yu et al., 2021), this is a first attempt to make instructions concise and engaging.

Original: Pay attention to weather forecasts. When planning a stargazing outing, you want to make sure you have good, clear weather. Cloudy skies will severely limit your ability to see stars. Additionally, stargazing requires spending long stretches of time outdoors, so you should avoid having to stand in the rain.
Rewrite: Ensure clear weather for your stargazing outing. Cloudless skies are key; check forecasts to avoid rainy disruptions.

Table 3: Example of rewrite for wikiHow steps

Results Since rewriting is an expensive process and we cannot send any task articles from private data sources to OpenAI, we limit our rewrites to a curated set of 14 wikiHow articles that are publicly available. These articles were shown to the user as suggested tasks (during the *Fitness Theme* weeks). Rewriting reduced the average number of words in a step from **55.92** to **27.94**, a $\sim$50% reduction in length. Through manual inspection, we ensured that no critical information was deleted during the rewrite.

This result rounds out contributions of this section, so we pause to summarize the key takeaways:

> *Our novel generation strategies allow for shorter, more engaging, more human-like responses, while maintaining the safety and efficiency of symbolic approaches.*

5 Community-Centric Multimodal Design

To ensure our system provides a benefit for AAVE speakers and signers, it is vital that we reach out to members of these communities in order to assess their needs and desired features for our system.

5.1 Co-Designing with African American Vernacular English Speakers

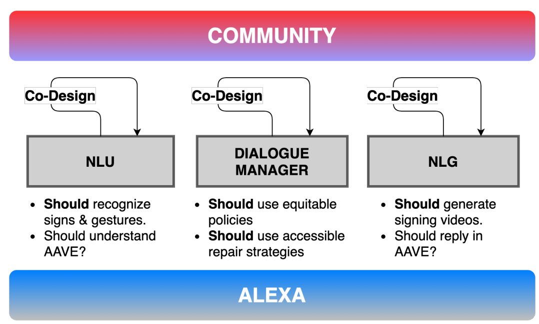

Figure 5: This diagram shows the community centric approach that we take to design ISABEL. We engage African-American Vernacular English speakers and American Sign Language signers.

We embark on a mission to champion inclusivity in voice assistance devices for a targeted audience of 30 million individuals whose first language is not English. We recognize the significance of accessibility for these users and the adverse effects of voice assistance devices that fail to support their native language. By testing of our voice assistance device with African American Vernacular English (AAVE) speakers in a pre-survey phase, we uncovered valuable insights concerning AAVE words like "Tryna" and "Finna," which were misunderstood or not understood by the system during dialogue processing. This testing motivated our team to go back to our past user data and find AAVE data sets for the analysis of code switching, non-understanding, and misunderstanding. While some aspects could be studied through user data (see Section 2.3), important aspects of the communities wants/needs cannot be inferred from data. Thus, we are empowered to create a survey to gather insight on this population's user experience and desires.

As most computational linguistics researchers traditionally focus on Natural Language Generation (NLG) to address such issues, our preliminary investigation delves deeper, aiming to explore the perceptions of AAVE speakers regarding voice assistance devices based on their past experiences and future desires. We propose focus on natural language understanding (NLU) first as the foundational step towards creating truly inclusive designs for voice assistance devices. We highlight work that is centered around the grammatical features of AAVE, challenges when using AAVE with technologies, and reports of the digital divide between AAVE and current technologies. To understand the perspective of this community, we explore a primary research question: How do African American Vernacular English (AAVE) speakers perceive and interact with voice assistance devices?

AAVE linguistic structure The challenges of integrating AAVE into Natural Language Processing (NLP) systems are evident in the face of its unique linguistic structure. For instance, AAVE exhibits variations in grammar such as habitual B and vocabulary such as "gonna, finna, wanna, tryna," (Green, 2002; Rickford, 2016) making it less compatible with traditional NLP models designed primarily for standard English.

AAVE user experience with devices Some of the main challenges faced by NLP systems when interacting with AAVE speakers are code-switching, misunderstanding, and non-understanding. AAVE speakers frequently shift between AAVE and English when the system fails to adjust to the diverse linguistic expressions used by the speaker (Benner et al., 2021; Harrington et al., 2022; Nwogu et al., 2023). When the NLP system cannot understand AAVE, it may respond with a non-understanding error message (Benner et al., 2021; Nwogu et al., 2023) or a misunderstanding caused by the system retrieving the next set of words that sound similar to the user's initial utterance (Benner et al., 2021; Nwogu et al., 2023).

Method To position our research team to think about solutions to support diverse identities creatively, we participated in a team activity (a vignette survey) that gives different scenarios, reflecting diverse, real-world user experiences, and asks the team to think about how ISABEL would perform in these situations. After this activity, we proposed to focus on AAVE users and their challenges when interacting with voice assistance devices. Our team's initial investigation highlights the need to involve this community in our design process, and we have begun initial outreach. We are working on a formal community survey to better understand and address this community's needs, meanwhile, Figure 5 highlights some key questions we aim to address through this survey.

5.2 Co-Designing with the American Sign Language Community

ISABEL, for the first time among the Alexa Prize bots, reached out to and incorporated the Deaf and Hard of Hearing (DHH) community through our partners at the prestigious Gallaudet University, which is chartered for the education of the DHH community. We have incorporated feedback from signers into the design process of our bot. For the first time, our bot supports giving sign language instructions using videos, and we have deployed a working model accessible to all public users.

For DHH community outreach, we have designed a survey, and we have reached out to a number of students and their families at Gallaudet University. A team from Gallaudet University led by Prof. Lorna Quandt has been involved in advising us since the beginning of the competition. Some of the feedback we have incorporated from the survey into our design process includes: considering the cognitive load of signers, changing the dimensions of the text, video, and images used to communicate instructions, as well as changing the design of the interface for ASL signers.

6 Evaluation

At the onset of our design for ISABEL, we set out to build an *inclusive*, *human-like*, and *safe* dialogue system. On each of these points, we've found empirically measurable successes. We discuss these next.

Inclusivity Through an iterative co-design process we've identified key design elements that ASL signers and AAVE speakers want (and do not want) in multimodal interactive systems. Direct community outreach reveals ASL signers do want systems to communicate using ASL videos, while use of team-internal activities show the need to better understand how ISABEL can best serve the AAVE community. Using these insights, to improve our system for ASL signers, we've built a first-of-its-kind multimodal interactive system that allows signers to participate in a task-oriented dialogue using solely visual and touch-controlled forms of communication. To improve our system for AAVE speakers, we've begun community outreach and design of a formal survey to understand this communities specific needs. Motivated by the team's initial investigations, our system design for ISABEL, in particular, our context-specific intent modeling approach, promotes more equitable understanding of AAVE user utterances – 36% more than baselines.

Human-Likeness We've used discourse theory to design context-specific intent models, which improve upon non-contextual baselines by up to 44% precision and 91% recall. We've used contextual grounding strategies to correct ASR errors, allowing us to correct 32% of errors in real time and 60% of errors in post-hoc analyses. We've also implemented repair strategies such as clarification questions. Finally, our neurosymbolic generation and rewriting strategies allowed for shorter, more engaging responses from a symbolically planned dialogue system. Evaluation of responses reveal they are as engaging as human responses 100% of the time, and only categorized as repetitive 30% of the time. In case of rewriting, our techniques also allowed for a 50% reduction in word count without losing important information.

Safety Using distillation of LLMs, we've designed a safe and efficient retrieval system for suggesting tasks and communicating task instructions to the user. We've generated 2246 high-precision safety keywords across 8 categories, and we estimate our system retrieves better task results than baselines in 24% of cases.

7 Future Work

The novel architecture and components of ISABEL provide a solid base for the larger research community to continue our work, developing safer, more inclusive, and more human-like task-oriented dialogue systems. We elaborate on some future avenues of research below.

Dialogue Systems Tailored for DHH Community While our work provides a first step towards ASL generation in multi-modal task-oriented dialogue, greater work is needed to expand our pipeline. Larger databases with more ASL tasks will help reach more users and test the generalizability of our pipeline. Continued co-design with members of the ASL community is paramount for any future works.

Error Recovery Our novel contextual grounding techniques for ASR and SLU error recovery have proven successful in real-time deployment and post-hoc evaluation, but future work can still improve these. For example, external data sources like large language models can continue to improve our techniques, using word sequence probabilities as supplemental information.

Persuasion Our neurosymbolic strategies offer solutions for generating more engaging and human-like text to promote increased user interaction and satisfaction. Still, these superficial approaches are not always successful at engaging users. One issue that often arose with TaskBot users was a misalignment between the user's goals and the TaskBot's capabilities. As a hypothetical example, a user might want to engage in chit-chat style conversation about a sports team or other entities. In the future, more sophisticated dialogue management strategies can help ISABEL to better align user goals with its capabilities. For example, persuasion strategies (Joshi et al., 2021; Yang et al., 2021; Tran et al., 2022) can improve user-retention rates, increase user satisfaction, and promote increased task-completion. Our categorization of non-task related user intentions (Appendix J) provides a good first step in building these strategies for persuasion.

Safe Dialogue Generation Finally, our novel framework for safe generation utilizes multiple data sources to identify potentially unsafe keywords in user requests, task titles, and task instructions. Still, more fine-grained safety measures are needed, considering the surrounding context of the keyword as well, to reduce false positives. One promising avenue is article re-writing, in which substitution or removal of inappropriate content is done where possible. Articles that can be re-written to be safe are likely to be false positives, which are acceptable to users with a more careful choice of words.

References

Eugene Agichtein, Michael Johnston, Anna Gottardi, Cris Flagg, Lavina Vaz, Hangjie Shi, Desheng Zhang, Leslie Ball, Shaohua Liu, Luke Dai, Daniel Pressel, Prasoon Goyal, Lucy Hu, Osman Ipek, Sattvik Sahai, Yao Lu, Yang Liu, Dilek Hakkani-Tür, Shui Hu, Heather Rocker, James Jeun, Akshaya Iyengar, Arindam Mandal, Saar Kuzi, Nikhita Vedula, Oleg Rokhlenko, Giuseppe Castellucci, Jason Ingyu Choi, Kate Bland, Yoelle Maarek, and Reza Ghanadan. 2023. Alexa, let's work together: Introducing the second alexa prize taskbot challenge. In *Alexa Prize TaskBot Challenge 2 Proceedings*.

Malihe Alikhani. 2020. *Multimodal Communication: Commonsense, Grounding and Computation*. Ph.D. thesis, Rutgers The State University of New Jersey, School of Graduate Studies.

Malihe Alikhani, Baber Khalid, and Matthew Stone. 2023. Image–text coherence and its implications for multimodal AI. *Front. Artif. Intell.*, 6:1048874.

Malihe Alikhani, Sreyasi Nag Chowdhury, Gerard de Melo, and Matthew Stone. 2019. CITE: A corpus of image-text discourse relations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 570–575, Minneapolis, Minnesota. Association for Computational Linguistics.

Malihe Alikhani and Matthew Stone. 2019. "caption" as a coherence relation: Evidence and implications. In *Proceedings of the Second Workshop on Shortcomings in Vision and Language*, pages 58–67, Minneapolis, Minnesota. Association for Computational Linguistics.

Vidia Anindhita and Dessi Puji Lestari. 2016. Designing interaction for deaf youths by using user-centered design approach. In *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, pages 1–6. IEEE.

Nicholas Asher and Alex Lascarides. 2003. *Logics of conversation*. Cambridge University Press.

Katherine Atwell, Sabit Hassan, and Malihe Alikhani. 2022. APPDIA: A discourse-aware transformer-based style transfer model for offensive social media conversations. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6063–6074, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.

Dhanush Bekal, Ashish Shenoy, Monica Sunkara, Sravan Bodapati, and Katrin Kirchhoff. 2021. Remember the context! asr slot error correction through memorization. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 236–243. IEEE.

Dennis Benner, Edona Elshan, Sofia Schöbel, and Andreas Janson. 2021. What do you mean? A Review on Recovery Strategies to Overcome Conversational Breakdowns of Conversational Agents. [Online; accessed 21. Jul. 2023].

Danielle Bragg, Meredith Ringel Morris, Christian Vogler, Raja Kushalnagar, Matt Huenerfauth, and Hernisa Kacorri. 2020. Sign Language Interfaces: Discussing the Field's Biggest Challenges. In *CHI EA '20: Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–5. Association for Computing Machinery, New York, NY, USA.

Cindy Chambers. 2020. Mindfulness and Interpreter Cognitive Load. *Digital Commons@WOU*.

Khyathi Raghavi Chandu, Yonatan Bisk, and Alan W Black. 2021. Grounding 'grounding' in NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4283–4305, Online. Association for Computational Linguistics.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Herbert H Clark and Susan E Brennan. 1991. Grounding in communication.

Emily Dinan, Gavin Abercrombie, A. Stevie Bergman, Shannon L. Spruit, Dirk Hovy, Y-Lan Boureau, and Verena Rieser. 2021. Anticipating safety issues in e2e conversational ai: Framework and tooling. *ArXiv*, abs/2107.03451.

P. T. Petri Du Toit. 2017. Mitigating the cognitive load of South African Sign Language interpreters on national television. [Online; accessed 20. Jul. 2023].

Manaal Faruqui and Dipanjan Das. 2018. Identifying Well-formed Natural Language Questions. In *Proc. of EMNLP*.

Abraham Glasser, Kesavan Kushalnagar, and Raja Kushalnagar. 2017. Deaf, Hard of Hearing, and Hearing Perspectives on Using Automatic Speech Recognition in Conversation. In *ASSETS '17: Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 427–432. Association for Computing Machinery, New York, NY, USA.

Abraham Glasser, Vaishnavi Mande, and Matt Huenerfauth. 2020. Accessibility for Deaf and Hard of Hearing Users: Sign Language Conversational User Interfaces. In *CUI '20: Proceedings of the 2nd Conference on Conversational User Interfaces*, pages 1–3. Association for Computing Machinery, New York, NY, USA.

Lisa J. Green. 2002. *African American English: A Linguistic Introduction*. Cambridge University Press, Cambridge, England, UK.

Christina N. Harrington, Radhika Garg, Amanda Woodward, and Dimitri Williams. 2022. "It's Kind of Like Code-Switching": Black Older Adults' Experiences with a Voice Assistant for Health Information Seeking. In *CHI '22: Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–15. Association for Computing Machinery, New York, NY, USA.

Sabit Hassan and Malihe Alikhani. 2023. D-CALM: A dynamic clustering-based active learning approach for mitigating bias. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5540–5553, Toronto, Canada. Association for Computational Linguistics.

Sabit Hassan, Shaden Shaar, Bhiksha Raj, and Saquib Razak. 2018. Interactive evaluation of classifiers under limited resources. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 173–180.

Yulan He and S. Young. 2003. A data-driven spoken language understanding system. In *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No.03EX721)*, pages 583–588.

Julie Hunter, Nicholas Asher, and Alex Lascarides. 2018. A formal semantics for situated conversation. *Semantics and Pragmatics*, 11:10–EA.

Mert Inan, Piyush Sharma, Baber Khalid, Radu Soricut, Matthew Stone, and Malihe Alikhani. 2021. COSMic: A coherence-aware generation metric for image descriptions. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3419–3430, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mert Inan, Yang Zhong, Sabit Hassan, Lorna Quandt, and Malihe Alikhani. 2022. Modeling intensification for sign language generation: A computational approach. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2897–2911, Dublin, Ireland. Association for Computational Linguistics.

Rishabh Joshi, Vidhisha Balachandran, Shikhar Vashishth, Alan Black, and Yulia Tsvetkov. 2021. Dialograph: Incorporating interpretable strategy-graph networks into negotiation dialogues.

Baber Khalid, Malihe Alikhani, Michael Fellner, Brian McMahan, and Matthew Stone. 2020a. Discourse coherence, reference grounding and goal oriented dialogue. In *Proceedings of the 24th Workshop on the Semantics and Pragmatics of Dialogue*.

Baber Khalid, Malihe Alikhani, and Matthew Stone. 2020b. Combining cognitive modeling and reinforcement learning for clarification in dialogue. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4417–4428, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mahnaz Koupaee and William Yang Wang. 2018a. Wikihow: A large scale text summarization dataset. *CoRR*, abs/1810.09305.

Mahnaz Koupaee and William Yang Wang. 2018b. Wikihow: A large scale text summarization dataset.

Manfred Krifka. 2011. Questions. In *Semantics: An international handbook of natural language meaning*, volume 1. Walter de Gruyter.

Haochen Li, Chunyan Miao, Cyril Leung, Yanxian Huang, Yuan Huang, Hongyu Zhang, and Yanlin Wang. 2022. Exploring representation-level augmentation for code search. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4924–4936, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Brooke Macnamara. 2012. Interpreter Cognitive Aptitudes. *Journal of Interpretation*, 19(1):1.

Amit Moryossef, Kayo Yin, Graham Neubig, and Yoav Goldberg. 2021. Data augmentation for sign language gloss translation. In *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*, pages 1–11, Virtual. Association for Machine Translation in the Americas.

Lisa Nakamura. 1995. Race in/for cyberspace: Identity tourism and racial passing on the internet. *Works and Days*, 13(1-2):181–193.

Jennifer Nwogu, Amanda Buddemeyer, Rosta Farzan, Angela E. B. Stewart, and Erin Walker. 2023. Comic-boarding with Children: Understanding the use of Language in Human-Human and Human-Agent Dialogue. In *IDC '23: Proceedings of the 22nd Annual ACM Interaction Design and Children Conference*, pages 667–671. Association for Computing Machinery, New York, NY, USA.

Andrew Perrin. 2022. Mobile Technology and Home Broadband 2021. *Pew Research Center: Internet, Science & Tech*.

Kun Qian, Satwik Kottur, Ahmad Beirami, Shahin Shayandeh, Paul Crook, Alborz Geramifard, Zhou Yu, and Chinnadhurai Sankar. 2022. Database search results disambiguation for task-oriented dialog systems. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1158–1173, Seattle, United States. Association for Computational Linguistics.

Arushi Raghuvanshi, Vijay Ramakrishnan, Varsha Embar, Lucien Carroll, and Karthik Raghunathan. 2019. Entity resolution for noisy ASR transcripts. 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): System Demonstrations*, pages 61–66, Hong Kong, China. Association for Computational Linguistics.

John R. Rickford. 2016. Labov's contributions to the study of African American Vernacular English: Pursuing linguistic and social equity. *J. Sociolinguistics*, 20(4):561–580.

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1668–1678.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Anthony Sicilia and Malihe Alikhani. 2022. Leather: A framework for learning to generate human-like text in dialogue. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 30–53.

Anthony Sicilia and Malihe Alikhani. 2023. Learning to generate equitable text in dialogue from biased training data. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2898–2917.

Anthony Sicilia, Katherine Atwell, Malihe Alikhani, and Seong Jae Hwang. 2022a. Pac-bayesian domain adaptation bounds for multiclass learners. In *Uncertainty in Artificial Intelligence*, pages 1824–1834. PMLR.

Anthony Sicilia, Jennifer C Gates, and Malihe Alikhani. 2023. How old is gpt?: The humbel framework for evaluating language models using human demographic dat. *arXiv preprint arXiv:2305.14195*.

Anthony Sicilia, Tristan Maidment, Pat Healy, and Malihe Alikhani. 2022b. Modeling non-cooperative dialogue: Theoretical and empirical insights. *Transactions of the Association for Computational Linguistics*, 10:1084–1102.

Elisabet Tiselius. 2018. Exploring Cognitive Aspects of Competence in Sign Language Interpreting of Dialogues: First Impressions. *HJLCB*, (57):49–61.

Nhat Tran, Malihe Alikhani, and Diane Litman. 2022. How to ask for donations? learning user-specific persuasive dialogue policies through online interactions. In *Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization*, UMAP '22, page 12–22, New York, NY, USA. Association for Computing Machinery.

Carla Viegas, Mert İnan, Lorna Quandt, and Malihe Alikhani. 2022. Including Facial Expressions in Contextual Embeddings for Sign Language Generation. *arXiv*.

Katinka Waelbers. 2009. From assigning to designing technological agency. *Human studies*, 32:241–250.

Haoyu Wang, Shuyan Dong, Yue Liu, James Logan, Ashish Kumar Agrawal, and Yang Liu. 2020. ASR Error Correction with Augmented Transformer for Entity Retrieval. In *Proc. Interspeech 2020*, pages 1550–1554.

Gabriella Wojtanowski, Colleen Gilmore, Barbra Seravalli, Kristen Fargas, Christian Vogler, and Raja Kushalnagar. 2020. "Alexa, Can You See Me?" Making Individual Personal Assistants for the Home Accessible to Deaf Consumers. *California State University, Northridge.*

Walt Wolfram. 2020. Urban african american vernacular english. In Bernd Kortmann, Kerstin Lunkenheimer, and Katharina Ehret, editors, *The Electronic World Atlas of Varieties of English*.

Runzhe Yang, Jingxiao Chen, and Karthik Narasimhan. 2021. Improving dialog systems for negotiation with personality modeling.

Kayo Yin, Kenneth DeHaan, and Malihe Alikhani. 2021a. Signed coreference resolution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4950–4961, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. 2021b. Including Signed Languages in Natural Language Processing. *arXiv*.

Kayo Yin and Jesse Read. 2020. Better Sign Language Translation with STMC-Transformer. *arXiv*.

Ping Yu, Yang Zhao, Chunyuan Li, and Changyou Chen. 2021. Rethinking sentiment style transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1569–1582, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*.

Xiaozhou Zhou, Ruying Bao, and William M. Campbell. 2022. Phonetic Embedding for ASR Robustness in Entity Resolution. In *Proc. Interspeech 2022*, pages 3268–3272.

Appendix

A More Details about Under-specified Intents

Navigation Intents Table 4 provides more details about the navigation intents for the task view that are introduced in Figure 3, including 1) a description of how the intent is triggered and the desired result and 2) the discourse relation between the user's turn that triggered the intent and the resulting actions of the bot.

Intent	Description	Discourse Relation
Back	User says a phrase or presses a button to go back	Acknowledgment
Step Details	User selects step (via voice or by clicking the step) to view more details	Elaboration
Resume	User says "continue" to move from video to steps	Continuation
Next	User says "next" (or similar) or presses a button to hear/view the next step	Continuation
Previous	User says "back" (or similar) or presses a button to hear/view the previous step	Continuation
Repeat	User says "repeat" (or similar) to hear the bot repeat the step	Continuation
Restart Instructions	User says "start over" (or similar) to restart the instructions	Continuation
Start Cooking	User says "start cooking" or presses a button to reveal the recipe steps	Continuation

Table 4: More information about the navigation intents introduced in Figure 3. The description outlines user actions that may trigger the intent, and the discourse relation refers to the relation between the user's turn that triggered the intent and the resulting actions of the bot.

Q/A Intents If the bot is able to, it will attempt to answer user questions. However, the type of information the user appears to be seeking will influence how the bot will attempt to answer these questions. Table 5 illustrates the different question intents, each of which maps to a different function.

Intent	Description	Resulting Discourse Relation
Query	User appears to be presenting a new query	Q/A
Grounding	User is looking for specific information about the task ("where is olive oil used?")	Q/A
General question	User is asking a general question while in a task ("how do you convert cups to liters?")	Q/A

Table 5: More information about the Q/A intents introduced in Figure 3. The description outlines user actions that may trigger the intent, and the discourse relation refers to the relation between the user's turn that triggered the intent and the resulting actions of the bot. For user questions, the resulting discourse relation will always be Q/A.

B Intent Classifier Details

B.1 Help Classifier

In the following cases our bot concludes that the user needs help:

- If the interaction model has already determined that the user intent is "HelpIntent". This rarely occurs based on our results.
- If the intent is not "HelpIntent" but the associated information to handle the predicted intent is unavailable. In such a case, our bot updates the intent to "HelpIntent". For example, if the intent is "SelectIntent" but the value of the selected item is "None", then the bot cannot proceed with the request, as the information to handle "SelectIntent" is absent.

Utterance	Domain	Search Query
how to fix a radiator	DIY	*fix a radiator*
cashew recipes	Cooking	*cashew*
recipe for cookies without sugar	Cooking	*cookies without sugar*
learn how to ride a bike	DIY	*ride a bike*
train a dog how to sit	DIY	*train a dog how to sit*

Table 6: Example queries (not real user utterances), and resulting queries extracted for search.

- If the user asks a question but the user's utterance doesn't have a noun phrase. The bot can determine if the user utterance is a question by checking if the intent is "QAIntent" or if the question classifier returns positive for the utterance.

Dataset for Evaluation We evaluated the Help Classifier using a random sample of real user data. Among these data points, we designed the dataset so 50% were instances where the bot served a help message (bot-positive occurrences), and the other 50% were instances where the bot did not provide help (bot-negative occurrences). Human annotators (from the team) reviewed all occurrences based solely on user utterances to establish the ground truth of "help" needed or other (i.e., "not help").

The bot-positive occurrences were evenly distributed across various triggers for providing help. Bot-negative occurrences were randomly sampled from the set of occurrences where the bot did not think that help was requested. We also ran these occurrences against Amazon's interaction model to obtain the intent classified by the interaction model. We then collated the results and evaluated the precision and recall.

As noted in the main text, other datasets for evaluating our intent classifiers were constructed in a similar manner.

B.2 Query Extraction

We used several strategies for extracting search queries from users' utterances. For utterances tagged as query intents by Amazon's interaction model, we used the slot values detected by the interaction model. Otherwise, we used our domain classifier and applied different query extraction strategies depending on whether the query was judged to be a cooking or a how-to task. If the query was classified as a cooking task, we used Spacy's noun phrase tagging and a list of common prepositions to isolate either a single noun phrase or (if applicable) multiple noun phrases attached with prepositions. If the query was classified as a how-to task, we isolated the query using a series of "pivot verbs", verbs that commonly appear at the beginning of a how-to article title in our dataset. We isolated the query by splitting on the last pivot verb that appeared in an utterance (or, if there is a noun phrase between the last two pivot verbs, we split on the second-to-last pivot verb) and take the part after (and include) this pivot verb as our query. In Table 6, we provide sample user utterances and the resulting extracted queries.

B.3 List Item Selection

B.3.1 Method

Once our bot extracts a query from user input, it returns the top three items from the search results from which users select one of them to start a task. Our bot decides which item is selected based on the user event on APL devices, the selection intent from the interaction model, and the user's utterance.

User event & selection intent The user event and the selection intent explicitly tell which option is selected. Still, a user could say, "Select the 3rd item" after scrolling the list to the right on APL devices, so that item could actually be, for example, the 6th item in the list. In this case, we count from the first item displayed on the devices.

Understanding the users' utterances If a user talks to our bot and the interaction model fails to recognize their correct selection intent, our bot tries to understand their utterances in three steps.

		Select Intent	Other intents
Interaction Model Only	**P**	1.00	0.67
	R	0.21	1.00
Interaction Model + Our Model	**P**	0.74	0.98
	R	0.97	0.79

Table 7: Precision (**P**) and Recall (**R**) for the select intent and other intents predicted by the interaction model only and the interaction model and our model combined.

1. See whether an item that fuzzy matches the whole utterance exists. The threshold was 96%.
2. Look for keywords that indicate a selection (e.g., "2nd", "second", "2", and "two" for the second item).
3. Check if the utterance contains one of the items' names.

We repeat these three steps for slot values if the interaction model classifies the utterance as a query or correction on our bot's response by a user.

If the three steps do not work, our bot checks if any noun phrases in the user utterance are contained by exactly one option. For instance, if the available options are Smoked Texas BBQ Brisket, Grilled Stuffed Flank Steak, and Seared Bluefin Tuna Steaks and the user says "tuna steaks," then they probably meant to choose Seared Bluefin Tuna Steaks instead of starting a new search for tuna steaks. In this case, our bot asks the user to confirm the selection. However, if there are multiple options that contain a noun phrase in the user utterance, the bot does not select the options because the phrase is under-specified.

B.3.2 Statistics

We collected user interactions between March 28 and July 27, of which 10% successfully selected at least one task. 86% of all successful selections were the user events on APL devices. 7.6% contained keywords for selection or were classified as the selection intent by the interaction model. These together show users were less likely to specify options by expressing the names of recipes or tasks.

To evaluate our model, we randomly sampled user utterances labeled as the select intent, along with an equal number of utterances labeled as other intents (i.e., when the select intent was a valid option by our model). We manually annotated their intents and found 76% indeed intended to select an option. As shown in Table 7, the interaction model alone is very conservative about classifying user utterances as the select intent. Our model remedies this by using context to scan for a variety of ways to select an option and shows very high recall for the select intent while maintaining reasonable precision.

B.4 Step Navigation

Navigation intents are used to identify the different ways of navigating the task instructions presented by the bot. These include navigating between steps of a task ("previous" and "next"), repeating the steps, and resuming a task. Acting correctly on these intents is necessary for a smooth flow of the dialogue. We use a context-driven classifier to differentiate these intents. The design of this classifier is based on the extensive evaluation of the interaction data with the real Alexa TaskBot users.

The step navigation classifier handles the following user intents- StartOverIntent, StartCookingIntent, CompleteIntent, NextIntent, ResumeIntent, RepeatIntent, and PreviousIntent. The classifier uses a set of words (n-grams) for each of these intents as the initial comparison step in the classification. The list of grams is based on user utterances, and therefore, omitted to preserve user privacy. The presence of this sequence of words in the user utterance may indicate the presence of the matched intent, but an empty match depicts the absence of that particular intent. The following sequence of steps is followed in the navigation classifier:

- Initial sequence matching of user utterance with the grams.
- Classification to distinguish between navigation/question answering.
- Context-based filtering of intent matches to remove false positives based on noun phrases present in user utterance.

- If there are multiple intent matches, preference is given to higher priority intents.

Table 8 shows a detailed breakdown of the precision and recall values for various intents under step navigation based on human-annotated data samples.

		StartOver	StartCooking	Next	Resume	Repeat	Previous
W/O Context	**P**	1.0	0.85	0.76	0.77	1.0	0.57
	R	0.11	0.8	0.54	0.57	0.24	0.57
W/ Context	**P**	1.0	0.85	0.94	1.0	0.95	0.74
	R	0.22	0.8	0.93	1.0	0.48	0.6

Table 8: Detailed results for the navigation classifier. Precision and Recall for the navigation intents classification with and without context based on human-annotated conversation samples.

B.5 Stop Classifier

With an increase in user interactions and reviews, we observed various differences in how different users want to end their conversations with the bot. Based on these interactions, we developed a stop classifier to identify a user's intent to stop their conversation with the bot at any time during the dialogue.

The stop classifier follows the same design structure as the navigation classifier and is the first classifier (highest priority) that the user utterance is passed through for intent extraction. Since stop intent will affect the user experience the most, the word sequences are carefully curated from the user utterances to avoid any false positive and hence a negative user experience. The list of grams is based on user utterances, and therefore, omitted to preserve user privacy.

B.6 Back Classifier

We developed a context-based pattern-matching classifier to identify the "back" intent from the user utterance. The back classifier has a similar structure as the other sequence-matching classifiers (navigation and stop classifiers). This classifier helps identify the user intent to either go back to the landing screen or to the initial options screen.

This classifier is mostly triggered in the case of headless devices, as the intent is identified through user utterances only on headless devices; whereas, on multi-modal devices, users can use the "Back Button" to trigger the same behavior. Following a similar design as before, this classifier works in a series of steps:

- Sequence match with the word sequences for back intent.
- Identifying whether the utterance is fired from inside a task or on the options screen.
- Sending the appropriate sub-intent (back to options or landing screen) based on the above steps.

B.7 Theoretical Study of Intent Classifiers Efficiency

At a high level, the rule-based intent classifiers we use in our model can be broadly described by the logical form:

$$Y = (A_1 \vee A_2 \vee \ldots \vee A_k) \wedge \neg(R_1 \vee R_2 \vee \ldots \vee R_\ell) \tag{1}$$

where Y is the decision for the intent, each A_i is a possible "accept" conditional which signals presence of an intent, and each R_i is a necessary "reject" conditional which signals an intent *cannot* have occurred. For example, conditionals can include requirements on n-gram containment, noun phrase number, and state variables.

Sample Efficiency Using an Occam Bound as described in Shalev-Shwartz and Ben-David (2014) (Theorem 7.7), we can estimate the sample complexity of our intent classifiers, and some baselines, in the traditional PAC learning theoretic sense. Complexities are given in Table 9. In particular, PAC learning theory gives bounds on the expected generalization gap – or, the gap between train and test set performance. This type of evaluation is important when deploying and updating models in real-time, as training and testing data cannot always be perfectly independent. Lower PAC sample

	W/ Context	W/O Context	Neural
Complexity[2]	$O(\log V/m)$	$O(\log V^2/m)$	$O(P/m)$
Example m	100	200	10,000

Table 9: Sample complexity computed using Occam's Bound where V is the vocabulary size for the rule-based models and P is the parameter count for the neural model. Neural model complexity is computed using a discretization trick (Remark 4.1, Shalev-Shwartz and Ben-David (2014)). We assume the size of the context representation is much smaller than the vocabulary size for rule-based models. Example m shows the number of samples needed to estimate a fixed generalization gap of $\sqrt{0.03}$ for each model's sample complexity, ignoring constant terms. V is set to 1000 and P is set to 300. Models **W/O Context** have higher sample complexity because they typically require bi-grams (at least) whereas the contextual models we propose can use uni-grams to achieve a similar coverage.

Intent	Added Utterances
Start Over	3
No	17
Yes	25
Previous	11
Next	20
Resume	4
Select	13
Stop	4
Repeat	46
Greeting	10
Back to Landing	4
Back to Options	11
Step Grounding	22
Meaning Grounding	13
Ingredient Amount Grounding	26
Correction	23
Summary	29
Query	154
Other Amazon Defaults	-

Table 10: Count of Interaction Model Sample Utterances

complexity bounds can also correlate with the ability of a classifier to transfer to new data as well (Sicilia et al., 2022a), making it a vital calculation for shifting user populations (e.g., as can be experienced during advertising campaigns).

B.8 Interaction Model

The first layer for all of our intent classifiers is Amazon's Interaction Model, which offers a context-less determination of the intent – using only the user's utterance. It also serves as a baseline for evaluation without context. We populated this with human effort early in the competition, before developing our context-aware approaches. Sample sizes are shown in Table 10. We also used the distillation of large language models to populate these intents, which we discuss next.

B.9 Distillation of Large Language Models

A crucial step toward responding appropriately is to understand the user's intent. A user may ask for elaboration, correction, or ask to start a new task. We identify 10 such intents and ask GPT to generate examples of users expressing such intents. These generated examples are then manually verified and added to our interaction model.

We observe that while GPT generated very relevant examples for intents such as Long Answer or Greetings (>80% cases), it had difficulty for intents such as Grounding (60% cases).

Intent	Example of Generated Data
Long Answer	What is the best way to {task}? Could you tell me the recipe for {task}?
Set Timer	Can you set a timer for {duration}?
Elaboration	Could you go into more detail about {task}?
Correction	I think you missed a step there
Substitution	Is there a way to switch {sub-phrase}?
Grounding	Explain {item} in simple terms. What part of the task was {item} used for?
New task	Let's switch to {task}.
Greetings	Good morning!
Repeat	Could you repeat that?
Stop	I'm outta here!
Select	Go for {option}

Table 11: Data augmentation for intent recognition; examples generated by `text-davinci-003`

C Details on ASR Error Recovery

C.1 Detailed Methodology

Using Context to Resolve ASR Errors The suggestions ISABEL makes as well as n-grams from ISABEL's intent models provide simple list-like representations of the current dialogue context. We exploit this context in two ways: re-ranking the n-best ASR hypotheses by matching them with elements from the context and re-raking the context by looking for phonetic correspondence with the user's input.

1. **n-best ASR hypotheses** First, if the system does not understand the best ASR hypothesis (e.g., it cannot parse it), it then tries to understand another of the n-best ASR hypotheses, running the same NLU operations on each. If the NLU operation works on one of the other hypotheses, this hypothesis is considered the correct transcription.
2. **Phoneme matching** The first solution can fail if the correct transcript is not contained in the n-best ASR hypotheses. To remedy this, we also re-rank the context based on the length and coverage of the longest common *phoneme* subsequences between the user utterance and the elements of the context. Matching in phoneme space allows us to explore more diverse possibilities for ASR mistakes, of which even the ASR engine is not aware.

It is important to note that our first solution is based on ideas from He and Young (2003), while our second solution is based on ideas from Wang et al. (2020); Zhou et al. (2022). Our novel contribution comes from limiting the space for correction to the elements from the context to provide more precise correction.

Matching Against Large Contexts through Search While the approaches just discussed can be effective for smaller representations of context, like the lists of options our system displays to a user, other parts of the dialogue may be more open-ended. For example, user queries to initiate a task could be referring to one of the thousands of possible supported tasks. In this case, we take an approach similar to Raghuvanshi et al. (2019) and Bekal et al. (2021), assuming correct texts make more sense than erroneous texts and therefore will return better results (i.e., if we use the texts as input to a search algorithm over tasks). However, our situation is different from theirs in two ways. First, unlike named entities, there are multiple ways for users to express a particular task intention: e.g., "build a wooden fence" could be phrased as "make a wood fence" or "construct a wooden barrier." Second, we cannot afford a large deep-learning model during runtime due to latency. To tackle these constraints, we use a fast semantic search algorithm (discussed in Section 4.1) as external memory for the search results. Similarly to before, we re-ranked the n-best hypotheses and search results based on textual and phonetic correspondence. The detailed steps are as follows:

1. If the interaction model determines the intent of the user is to search for a query and this intent fits the context of the dialogue, for each n-best ASR hypothesis,
   (a) run the indexed search on the slot value and
   (b) rescore the search results by considering syntactic correspondence (e.g., word orders), too.
2. If any of the n-best ASR hypotheses do not exceed a certain threshold, run the indexed search with their phoneme sequences.

The preconditions in step 1 help us avoid over-correction of user inputs because this method works only with search queries and the interaction model gives us the interval of the query slot value. Step 1b is crucial to avoid false acceptance of a hypothesis that uses almost the same set of words but conveys a different meaning.

C.2 Statistics on the ASR Error Dataset

45% of the ASR errors considered in the main text are not addressed by our proposed approach. The top reason (28%) for the failure of our method is that the user input was not classified as the query intent, and therefore our method was not triggered at all. In 22% of unaddressed errors, the audio input was cut off so early that the transcript had no information on what the user wanted to do. On the other hand, 10% of them seem to have a complete transcript, but human annotators still could not resolve the error, using the given context. 13% happened when a user asked a question about their task, while 7% happened when they asked a question not related to ISABEL. ASR errors also happened when a user did not know what to say to ISABEL (11%).

C.3 Detailed Error Analysis

Error Analysis for RQ1 60% of errors not corrected occurred when users wanted to select an option or followed a query suggested by ISABEL, and most of them had more than 60% phonemes in common with the correct context (slightly below our threshold). The rest happened while users were navigating through tasks and tended to have low matching percentages because the lengths of keywords are short and therefore one error impacts the percentages greatly. The dialogues where ASR errors were corrected at least once in real-time had a statistically significantly higher rating ($M = 4.0, SD = 1.3$) than those where errors were not corrected ($M = 2.7, SD = 1.6$) with the t-test ($t = 3.9, p < 0.001$).

Half of the false positives happened during the selection of an option, and the rest happened during tasks. All the false positives during tasks misinterpreted the users' intent to ask a question about a task or start new a task as the intent to go to the next step. The ratings of the conversations with false positives ($M = 2.2, SD = 1.3$) did not significantly differ from the conversations with uncorrected errors ($t = -0.8, p = 0.4$).

Error Analysis for RQ2 We found that successful corrections usually occurred when the correct transcript appeared in the ASR system's n-best hypotheses (55% accuracy in this case). In general, we could not properly correct errors that did not have correct transcripts in their n-best hypotheses because, in most cases, some of the n-best hypotheses returned good results in the search, and therefore our method stopped before utilizing phonetic information.

D Question Answering Models

D.1 Supporting Task-Specific Question Answering

The Question-Answer (QA) responder for ISABEL is responsible for handling user utterances that have been classified as "questions" during the *Task View* phase of the dialogue. Note, this excludes questions which are classifier by a higher priority intent model (like, the Navigation intent classifier). Because of the nature of the dialogue, ISABEL could answer questions in two ways:

1. **Short answers:** the response to such questions is usually one or two sentences short and not multi-step

    Q. How to power on an iPhone?

    A. hold and press the power button

2. **Starting a new task:** the response to such questions is usually composed of multiple steps

    Q. How to fix a radiator?

    A. (a multistep process)

The QA responder primarily inspects the user intent to decide how to handle the user utterance:

- **Query Intent**: If the user intent is determined to be "Query" by an upstream classifier, then the bot probes a default Amazon QA engine to get a response for the user's question and then informs the user about the short answer if found. Based on whether a task is ongoing, it additionally asks the user a clarification question: *if the user would like to start a new task or not*.
- **Grounding Intent**: Certain questions are of very specific nature – answers to which can be extracted out from task instructions in a simple rule based manner, or can be answered by the default QA engine. For example:

    "What was the amount of salt needed in step 3?"

    "What does tbsp stand for?"

  Questions such as above are labelled by the Interaction Model as Grounding Intent. When possible (e.g., if answers can be easily found in the recipe or article), answers are generated using simple rule based approaches for analyzing articles. Otherwise, the question may aim to find the meaning of a particular item. These questions are parsed for the a "subject" and a re-phrasal – "What does subject mean?" – is passed to the default QA engine to try to generate an answer.
- **Fallback**: If none of the above cases are applicable then the bot does the following:

    1. Probe the default QA engine with user utterance

    2. If in step 1, and QA engine doesn't positively respond, the bot employs a noun phrase ranking strategy on the current task article to rank the steps and ingredients. ISABEL then presents the user with the top result (sorted based on the most number of noun phrase matches) with the user query. ISABEL is careful to clarify it didn't quite understand the question, but thinks this answer might help.

D.2 Linguistic Analyses Reveal a Long Tail in Distilled QA Data Sources

Many recent works have utilized data augmentation strategies to distill knowledge from a large language model (LLM) to a much smaller, less-computationally-intensive model. This technique has the advantage of yielding large training datasets without requiring a large human annotation effort. However, do these generated datasets differ in distribution from the test data, and how? One way to study this is by examining linguistic differences between generated data and the test set. We studied this in the context of question-answering: *can we make GPT generate a large dataset of question-answer pairs that reflect real user questions about our recipes?* To answer this question, we prompted GPT to generate 199,530 generated question-answer pairs based on Food.com recipes. We then used a rule-based classifier to determine the distribution of semantic question types in this dataset, and compared it to the distribution of questions asked by TaskBot users when shown a recipe (309 user questions in total).

We specifically focus on the semantic question types outlined in Krifka (2011): **polarity questions** (questions requiring a yes or no answer), **alternative questions** (questions that offer a choice between multiple alternatives), and **constituent questions** (wh-questions in English). We further split polarity questions into **declarative questions** (which syntactically resemble a declarative sentence) and non-declarative polarity questions, which we refer to as **Y/N** questions, in order to more accurately classify them using syntactic features.

We examine their distributions (Figure 6), and find key differences between the two. Namely, TaskBot users more commonly ask declarative questions, as well as questions that could not be classified by our classifier. We recognize that some of these differences may be due to classifier error. However, given that our classifier is rule-based and relies on syntactic features, these differences indicate, at the very least, syntactic differences between generated questions and user questions. Because

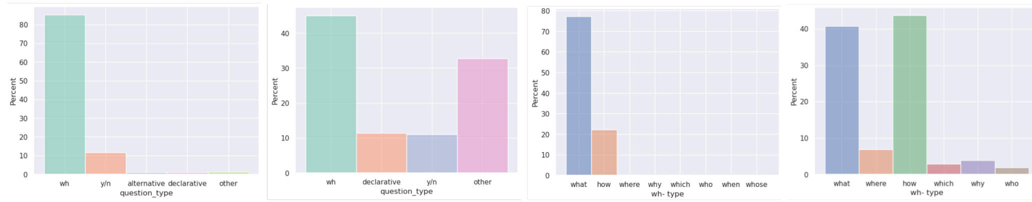

Figure 6: Distributions of (from left to right): semantic question types for GPT-generated questions, semantic question types for questions from TaskBot users, wh-questions for GPT-generated questions, and wh-questions from TaskBot users.

wh-questions made up a plurality of both generated and user questions, we further examined the distributions of wh-questions, according to which wh-verb they started with. The results, shown in Figure 6, indicate a distinct between GPT-generated wh-questions and TaskBot user wh-questions. However, both distributions reflect a long tail; a very low percentage of wh-questions from GPT and from real users start with words other than “what" and “how". This indicates potential difficulties that may occur when training and evaluating on a diverse set of wh-questions. Future work should further examine this long tail, and how distributional linguistic differences may impact the generalizability of knowledge distillation strategies on task-oriented dialogue agents.

E ASL Generation Details

Related Work for ASL Models for the cognitive aptitudes and cognitive loads of sign language interpreters have been studied before by Macnamara (2012); Du Toit (2017); Tiselius (2018); Chambers (2020). These models help guide the design principles of multimodal communication systems with sign language generation capabilities, as the user will need to focus on multiple modalities simultaneously through the bottleneck of a singular visual modality which induces cognitive load.

Work has been done to test Wizard-of-Oz systems where Alexa is combined with a camera to detect sign gestures in Wojtanowski et al. (2020).

Accessibility of personal assistant devices to the Deaf and Hard of Hearing community has been assessed multiple times before by Glasser et al. (2017, 2020); Bragg et al. (2020)

Design approaches incorporating the DHH community have been proposed before by Anindhita and Lestari (2016).

ASL Generation To generate sign language instructions, we employ the pipeline in Figure 4. We first retrieve instructions for a given task, and then we convert each step into GLOSS representation using rule-based sign language translation algorithms and also using Large Language Models. Afterward, we segment each instruction into separate gloss tokens and retrieve sign videos for each gloss token from `Lifeprint.com` and YouTube. We then store these videos in an S3 bucket, then during each step of the task, we retrieve the videos corresponding to each gloss token and stitch them back-to-back to create a continuous video sequence. We show this sequence of videos in addition to a picture of the step. The picture for each step generally shows the result of the action as described in the sign instructions.

F Background on AAVE and Challenges with Traditional NLP

The challenges of integrating African American Vernacular English (AAVE) into Natural Language Processing (NLP) systems are evident in the face of its unique linguistic structure. AAVE has a unique structure that differs from the standardized English used in current systems, leading to disparities in recognition and comprehension. For instance, AAVE exhibits variations in grammar such as habitual B, and vocabulary such as "gonna, finna, wanna, tryna," making it less compatible with traditional NLP models designed primarily for standard English (Green, 2002; Rickford, 2016). AAVE is well documented for its grammatical features but is considered a low-resource language in artificial intelligence due to the lack of training data and models that support it (Sap et al., 2019).

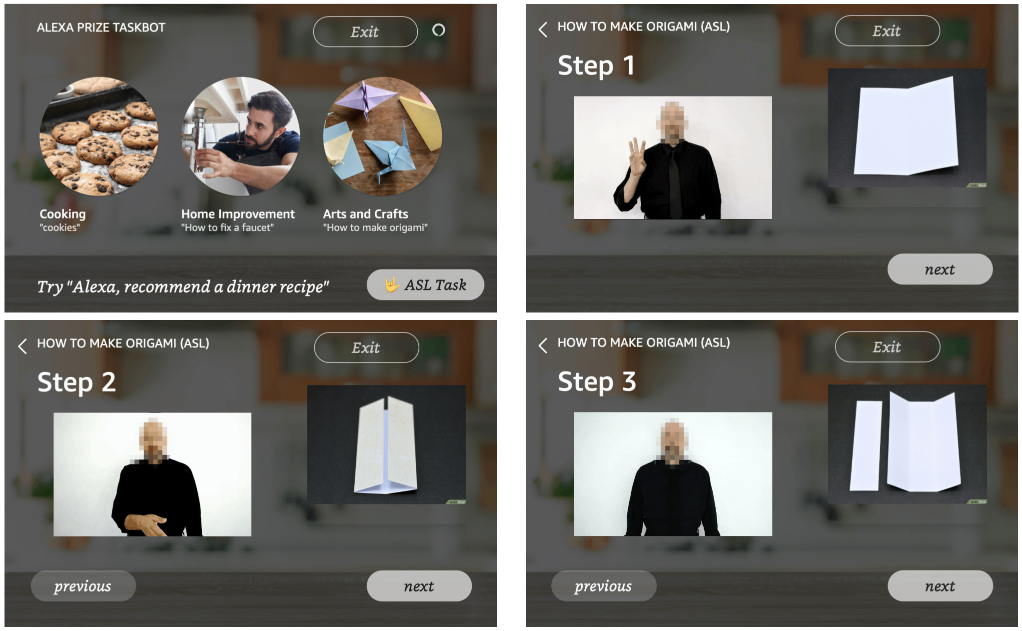

Figure 7: This figure demonstrates all the sections of a sample origami task with ASL video instructions.

Some of the main challenges faced by NLP systems when interacting with AAVE speakers are code-switching, misunderstanding, and non-understanding. AAVE speakers frequently shift between AAVE and English when the system fails to adjust to the diverse linguistic expressions used by the speaker (Benner et al., 2021; Harrington et al., 2022; Nwogu et al., 2023). Furthermore, when the NLP system cannot understand AAVE, it may respond with a non-understanding error message (Benner et al., 2021; Nwogu et al., 2023) or a misunderstanding caused by the system retrieving the next set of words that sound similar to the user's initial utterance (Benner et al., 2021; Nwogu et al., 2023). Studies show that this impact on identity can lead to significant alterations in an individual's technological agency (Nakamura, 1995). Technological agency refers to the power and control that individuals believe they have over technology, enabling them to modify, adapt, or even resist its use in ways that align with their unique needs and preferences (Waelbers, 2009).

Reports have highlighted the racial disparities in internet access and technology adoption. The Pew Research Center (Perrin, 2022) on the digital divide reveals how race and class impact internet access, with Black individuals and communities disproportionately facing barriers to connectivity. AAVE is spoken by 30 million individuals in the United States, and the lack of support for this language in voice assistance devices can contribute to the barriers this population is already facing in accessing and utilizing such technologies (Wolfram, 2020). To understand how this population perceives voice assistance devices, we conducted a preliminary survey with five questions.

G Details on Efficient Semantic Retrieval Algorithm

G.1 Algorithm Description

We first obtain relevant articles from a public Wikihow dataset (Koupaee and Wang, 2018a) by computing cosine similarity between embeddings of titles in this public dataset and any private task dataset(s). If cosine similarity between a pair is higher than a threshold, we add the public article to a set of candidates. We cluster these candidates and distill variations for the cluster centroids using GPT-3. As clustering has been shown to reduce bias in sample selection (Hassan et al., 2018; Hassan and Alikhani, 2023), we expect this to yield more diverse and representative candidates for generating variations. For a user query, we perform an indexed search on the distilled variations and return a mapping to the article title in the private dataset. Indexing allows for efficient search of the space.

Original	GPT-generated variations
serve tea	brew tea
	make a refreshing cup of tea
host a birthday party	plan a birthday party
	throw a birthday fiesta

Table 12: Examples of GPT-distilled variations for a larger search space.

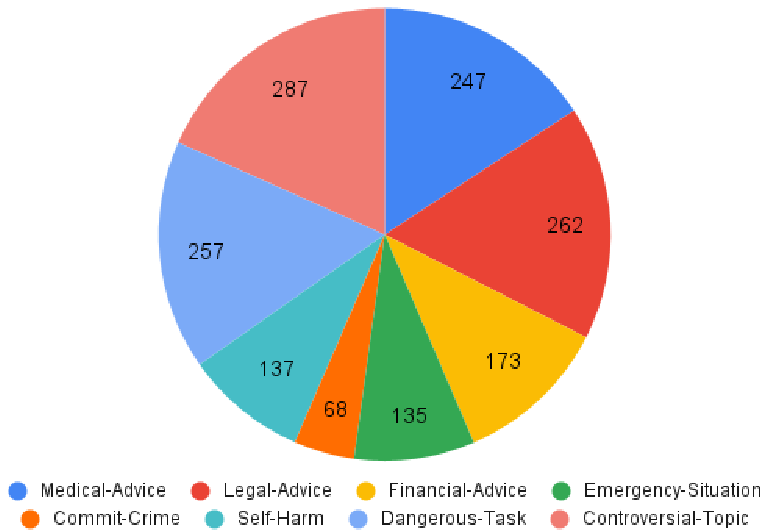

Figure 8: Distribution of our safety categories

G.2 Details on Results

We use (a private) MiniLM for obtaining Embeddings, Kmeans for clustering, and GPT-3.5-turbo for generating variations. We use 20K clusters with k=8, resulting in 160K additional variations.

H Data Sources for AI Safety

Knowledge Distillation In our initial exploration, we prompted both GPT (da-vinci-003) and ChatGPT (gpt-3.5-turbo) to generate 50 samples at a time for the first four categories from Table 2 with the following prompt: "Generate 50 examples of a user asking questions about [category] to a conversational agent." While both GPT and ChatGPT generated coherent and relevant responses, we observe that ChatGPT generated more specific responses such as "There's a gas leak in my house. What should I do?" while GPT may generate generic examples such as "How can I get help in an emergency." As such, we use ChatGPT to generate subsequent batches of data for the four categories. We also generate data for Controversial Topics and Dangerous Tasks in a similar way using ChatGPT, yielding a total of **655** examples in addition to the 200 generated by GPT.

Knowledge distillation and human effort Since OpenAI's API refuses to generate responses for committing crimes, we manually prompted ChatGPT to generate 100 examples of crimes. These examples were then annotated manually to form sentences. After removing duplicates and irrelevant examples, we end up with 68 instances.

Social media data with human revision: Similar to committing crimes, OpenAI's API doesn't allow the generation of self-harm data. Prompting to generate indicators of self-harm also leads to the model insisting that the prompter not cause self-harm, rather than generating the data. As such, we opt for the subreddit r\depression as our data source for self-harm. We obtained 100 posts and

	Max	Min	Average	Std
Sentence Length	20.00	6.00	11.03	2.10
Offensive rate	0.00%	0.00%	0.00%	-
well-formedness	0.9682	0.0870	0.5993	0.2032
readability	17.00	0.10	6.4487	3.1776

Table 13: Evaluation results of opening encouragement lines. Sentence length is the word count for each line. The metric offensive rate is adapted from Barbieri et al. (2020). The metric for well-formedness is adapted from Faruqui and Das (2018). An automated version of the Flesch–Kincaid readability test is used for our readability metric.

manually extracted key phrases suitable for our guardrail. This edit was necessary because comments were often too long. After filtering out very similar or irrelevant posts, we obtained 72 instances.

Keyword List In addition to the above, we obtained legal, medical, and financial terminology from multiple websites (e.g., https://www.uscourts.gov/glossary) to add relevant terms to our guardrails. We obtain a total of 553 terms this way. After extracting keywords from the rest of the dataset, we ended up with 2246 keywords. We also add keyword lists provided by Amazon to our list.

I Details on Neurosymbolic Generation Strategies

Below, we summarize some of the places where we use LLM generation, symbolic algorithms, and combinations of both techniques. In the next parts, we go into the details of each technique.

- **Opening Encouragement:** To inspire users to start tasks and stimulate enthusiasm to complete tasks, we use an LLM (gpt-3.5-turbo) to generate motivational opening remarks for all tasks within a publicly available wikiHow dataset from Koupaee and Wang (2018b). These remarks are tailored to stimulate user engagement. Automated analyses show these remarks are *short, not offensive at all, a 6th grader can understand them, and they are mostly well formed* (see Appendix I Table 13). Examples are provided in Table 14.
- **Progress Remarks:** Progress notes are also generated by an LLM to continue to stimulate user enthusiasm throughout the task. This process requires categorization by the LLM along with note generation. Examples of these notes are provided in Table 15.
- **Displaying Search Options:** When responding to user queries to start a task, the options retrieved by our search algorithms are communicated enthusiastically using LLM-generated responses. Symbolic algorithms are also used to compare the retrieved options and provide multimodal communication of this comparison to the user; e.g., adding short comparisons of time, rating, etc. to **ISABEL**'s speech output as well as emoji descriptors on the screen for visual communication. For additional technical details and results, see § 4.3.1.
- **Help Messages:** Help messages are used to guide users when they appear to be confused about what they can do, or **ISABEL** cannot understand them. LLM-generated partial utterances combined with a symbolic retrieval algorithm allow us to ensure diversity and enthusiasm in our help responses.

I.1 Opening Encouragement

To inspire users to start tasks and generate enthusiasm to complete tasks, we use an LLM (gpt-3.5-turbo) to generate motivational opening remarks for all tasks within a publicly available wikiHow dataset from Koupaee and Wang (2018b). These remarks are tailored to stimulate user engagement.

We carefully selected the appropriate prompt for generating these lines. We experimented with various prompts, using a sample of 100 tasks. Each generated line was then manually assessed based on several criteria, including sentence length, incidence of offensive language, syntactic correctness, readability, and motivational strength.

After a thorough examination, we settled on the following prompt to generate encouraging lines for the public dataset: “You are a motivational AI assistant whose goal is to inspire and encourage users to do the task steps in articles. Write an advertisement line so the users get excited to start the tasks in one sentence. Do so in less than 15 words. Start with getting ready to, or something similar."

Title	Encouragement Lines
How to Clean a Book	Keep your books clean and in top condition with expert cleaning techniques!
How to Use YouTube Uploader	Master the art of YouTube uploading and share your creativity with the click of a button!
How to Avoid Car Sickness	Say goodbye to car sickness and enjoy a comfortable journey with these simple methods!

Table 14: Examples of opening encouragement lines.

Category	Progress	Progress Note
Cars & Other Vehicles	10%	Buckle up! You've just started cruising through the steps. Keep that engine roaring!
Arts and Entertainment	30%	Step by step, you're turning the art of progress into a masterpiece.
Computers and Electronics	50%	You're halfway there! Keep rockin' those tech steps and stay ahead of the game!
Education and Communications	70%	Congratulations on your progress! You're getting closer to completing this educational journey.
Health	90%	You're almost there! Keep up the great work and pat yourself on the back!

Table 15: Examples of progress notes.

Several examples of the lines generated through this process are presented in Table 14.

I.2 Progress Remarks

Our strategy for producing the Language Model (LM) generated progress notes consists of two significant steps: task annotation and progress line generation for each category.

Firstly, we employ the GPT language model to categorize each article within the publicly accessible wikiHow dataset into one of the 20 predefined categories. The distribution and categorization of the dataset are illustrated in Figure 9. Once categorized, the annotated dataset is stored in S3 cloud storage for future reference. The prompt used for this step is: "You are an AI assistant that categorizes articles into groups given titles. Given the following article titles, use your language understanding capabilities to categorize each into an appropriate topic. “

After annotating the dataset, we generated 30 unique progress lines for each category. Additionally, we generated 30 progress lines without categorization information. These progress lines are further divided into ten steps, each containing three lines. The resulting 630 progress lines are stored locally within ISABEL for quick access. The prompt for generating these progress lines is: "You are a motivational AI assistant walking a user through the steps of a task. Generate a witty progress line that encourages the user and acknowledges their progress. Do so in less than 15 words. Don't mention the exact step numbers. Category: Current Step: Total number of Steps:"

Once a user starts a task with ISABEL, it attempts to find the corresponding title in the annotated dataset to identify the task's category. ISABEL then computes the user's current progress and gets three progress lines using the category information and the progress rate. Out of these three, it chooses to present a line randomly. ISABEL only shows the progress lines based on a probability that can be adjusted - currently set at 20%. A few examples of these progress lines are displayed in table 15.

I.3 Displaying Search Options

As noted in the main text, we use LLMs to generate an engaging message to display search options retrieved after a search. For example, when the user prompts ISABEL by asking “how to” do a certain task. We show some examples of the messages we generate in Table 16. While most of the structure and content of the message is generated by the LLM (discussed in Appendix I.4), some

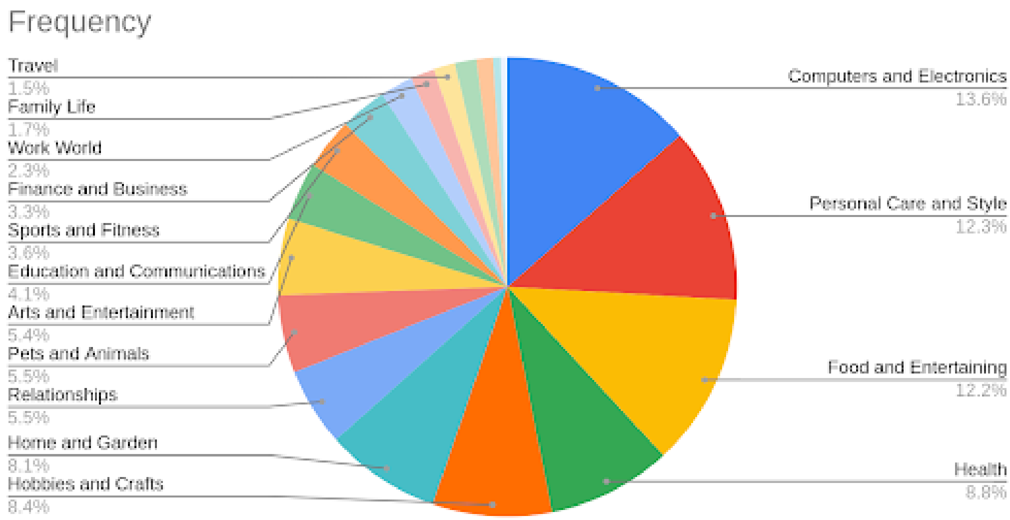

Figure 9: Distribution of annotated categories in wikiHow dataset.

important article information contained in this message is symbolically generated and presented in a multimodal fashion, using the procedures discussed next.

Comparisons and Emojis When users are provided results for their search query, several good candidates may match their query. However, some candidates may fit the user's needs more than others, and a list of results with task images does not always convey which candidates are most appropriate. For instance, a recipe search on Whole Foods may yield three or more recipes entitled Blueberry Muffins, and the best recipe for the user will depend on whether the user is most concerned about health, other users' opinions of the recipe, or the time the recipe takes to make. To provide the user with more information, we use two tools across different modalities: comparison and emojis. Incorporating both into ISABEL allows us to communicate more information to more users.

Comparison: When the user searches for a recipe, in addition to reading and displaying the top three search results, ISABEL provides a one-line description detailing some differences between the recipes. This description will vary based on a random selection between possible candidates, which include *the first, second, or third option is the highest rated* and *the first, second, or third one is our overall pick*. This keeps the descriptions short and prevents the ISABEL's utterances from becoming repetitive. Our comparison feature uses the speech modality for both headless and multimodal devices.

Emojis: Though our comparison features provide some information, we restrict the amount of information it offers to reduce the user's cognitive load; there is only so much information the user can listen to and retain in a single dialogue turn. However, the visual modality can be utilized to convey additional information. By placing emojis and one-to-two-word natural language descriptions underneath each search result, we provide the user with eye-catching details that distinguish the different search results and are retained on the screen. An example of emojis displayed in the recipe search results can be found in Figure 10.

I.4 Generation with Formal Grammar Rules

This section explores prompting techniques to seamlessly combine LLM-generated outputs with the symbolic components of **ISABEL**. As noted in the main text, there are key lexical, semantic, and structural constraints imposed on the LLM generation as a result. We highlight these important constraints:

1. a symbolic algorithm will process the users' next utterance, so clear communication of possible "next steps" to the user and how to trigger these steps is important;

Search	Generated Message Examples
cookies	Let's explore some delicious Whole Foods recipe options for your search on cookies! Take a look at what I found: The top results are Nutella cookies, Linzer cookies, and Spritz cookies. The third option looks like the best. Choose one to get started or summarize it if you're short on time. Delicious recipes await you! Take a look at what I found on the Whole Foods marketplace for your search on cookies: The top results are Nutella cookies, Linzer cookies, and Spritz cookies. If you want the highest rated, check out the second option. Whether you want to select an option, summarize it, or search for something else - the choice is yours!
cake recipes	Let's spice things up in the kitchen with these tasty Whole Foods recipes! Here's what I found for your search on cake: The top results are Carrot Cake Coffee Cake, Louisa's Cake, and Hummingbird Cake. Other users have rated the first option pretty high, and the first option has the fewest steps. Don't forget to select an option, summarize it, or search for something else. Let's explore some delicious Whole Foods recipe options for your search on cake! Take a look at what I found: The top results are Carrot Cake Coffee Cake, Louisa's Cake, and Hummingbird Cake. . Choose one to get started or summarize it if you're short on time.
how to fix a faucet	Let's see what we've got! Here are some options I found on WikiHow for fix a faucet: The top results are How to Fix a Leaky Faucet, How to Fix a Kitchen Faucet, and How to Fix a Leaky Shower Faucet. You can select an option, summarize an option, or search something else. Are you ready to dive into the world of DIY? Check out what I found on WikiHow for fix a faucet: The top results are How to Fix a Leaky Faucet, How to Fix a Kitchen Faucet, and How to Fix a Leaky Shower Faucet. You can select an option, summarize an option, or search something else.
how to make origami	These options seem like a blast! Check out what I found on WikiHow for origami: The top results are How to Make Origami, How to Make an Origami Pig, and How to Make an Origami Chair. You can select an option, summarize an option, or search for something else. Get ready to have some fun! Here's what I found on WikiHow for origami: The top results are How to Make Origami, How to Make an Origami Pig, and How to Make an Origami Chair. You can select an option, summarize an option, or search something else.

Table 16: Examples of generated messages that appear for different search queries. By using LLM-generated candidates for several components of our message, and varying which candidates are compared and how, we are able to make the bot more humanlike and stochastic while still ensuring it presents the user with accurate information.

2. programmatic insertion of outputs from the system's symbolic generation components must be possible, e.g., retrieved articles and attribution in the case of displaying search results;
3. any necessary components of the generation (e.g., content attribution) should be present;
4. and, the user's query cannot be provided to the (third-party) LLM to preserve user privacy.

As noted in the main text, we use *formal grammars* (Chomsky, 1956) to achieve the constraints imposed on LLM generations (i.e., by the symbolic plans laid out by ISABEL's generation modules). For example, these constraints are needed when generating an engaging message to display retrieved tasks after a search. A formal grammar is a model of language defined by the following components:

- **Terminals** a set which defines the base alphabet or vocabulary for the grammar – we modify this slightly to increase stochasticity, asking the LLM to generate these itself;
- **Variables** which can take on different combinations from the terminal alphabet;
- **Production Rules** which describe how to go from the **start** symbol (a special variable) to other variables, and then to the terminals, which in the end, should form a full utterance in the grammar.

An example prompt that uses this grammatical structure is provided below. Interestingly, gpt-3.5-turbo appears to have an understanding of how to construct utterances from grammar rules and our prompting strategy is able to exploit this.

Example Prompt: *(temp: 1, top p: 0.9, freq penalty: 0.25)*

I am going to describe a formal grammar and then ask you to generate text according to this grammar. To promote more diverse derivations from the grammar, we will modify the usual definition so that terminals can be stochastic. To achieve this behavior, a similarity symbol "∼" will mean you should self-prompt, so you are supposed to generate the terminal value on your own according to the provided prompt. For more traditional, deterministic terminal assignments in the grammar, we will use the equality "=" symbol. These terminals should be copied exactly.

The set of stochastic terminals is below. Remember, you should generate these according to the self-prompt:

- fun_message ∼ generate something exciting to hype a user up before they start working on a DIY task. For example, "Are you ready to get started" or "These options seem fun!" or "This looks like it will be a blast!"
- search_results ∼ generate a message to display WikiHow article options that were found on WikiHow. Make sure to attribute WikiHow and use the placeholder "___input_search___" and "__list_options__" to indicate what was searched and what was found, respectively. For example, "Here's what I found on the WikiHow for ___input_search___: __list_options__." Copy both placeholders verbatim.

The set of deterministic terminals is below. Recall, these should be copied exactly as written here:

- instructions = "You can select an option summarize an option or search something else."

The set of variables are:

- FUN_SEARCH
- INSTRUCT

The start symbol is:

- START

The set of production rules are:

- START -> FUN_SEARCH __instructions__
- FUN_SEARCH -> fun_message search_results
- FUN_SEARCH -> search_results fun_message

An example derivation is given below:

Mmm, these recipes look delicious! Here's what I found on the Whole Foods marketplace for __search__: __list_options__. You can select an option summarize an option or search for something else.

Generate 10 diverse and fun examples according to this grammar. Respond differently to the self-prompts each time. Use all the rules with equal likelihood.

Example Output:

- Are you ready for some DIY fun? Here's what I found on WikiHow for __input_search__: __list_options__. You can select an option, summarize an option, or search for something else.
- Oh boy, this is going to be exciting! I found some awesome ideas on WikiHow for __input_search__: __list_options__. You can select an option, summarize an option, or search for something else.
- These options seem like a blast! Check out what I found on WikiHow for __input_search__: __list_options__. You can select an option, summarize an option, or search for something else.
- Get ready to unleash your creativity! Check out these ideas I found on WikiHow for __input_search__: __list_options__. You can select an option, summarize an option, or search for something else.

- `Let's see what we've got! Here are some options I found on WikiHow for __input_search__: __list_options__. You can select an option, summarize an option, or search for something else.`

I.5 Help Messages

Help messages guide the users when they are lost. This means that they require context awareness, where we need to let the user know what actions they can perform based on the current state of the conversation. In order to do that we employ a set of grammatical rules and templates. They are based on the current state and the responder that is shown to the user.

Similar to the above section, we generate help messages based on formal grammatical rules. For help messages, rather than supplying rules *within* the prompt to an LLM like `gpt-3.5-turbo`, we instead use human effort to design the grammar's production rules and variables, automating only the generation of terminals (i.e., using `gpt-3.5-turbo` with some human supervision). The randomly generated terminals can then be combined using simple deterministic algorithms.

For example, one grammatical rule for generating help messages is the following:

`start` → `confused` + `eurekaIntj` + `info` + `whatWantDoQ`

where '+' denotes concatenation and all other terms are variables. `gpt-3.5-turbo` can then generate multiple terminals that map directly to each variable with human supervision; e.g.,

`confused` → Hmm, it looks like we need some help getting started.
`confused` → You don't know what to do, right?
`confused` → I'm having trouble understanding you.
...

After randomly generating terminals, we can execute rules with a simple deterministic algorithm. So, a full help message corresponding to the rule from before could look like:

> Hmm, it looks like we need some help getting started. Aha! You can ask me to help you make a recipe or do a DIY task. What would you like to do today?

A high degree of stochasticity comes from having `gpt-3.5-turbo` generate multiple potential terminal values for each variable, making the help messages appear more human-like.

J Details on User Requests from Taskbot

From analysis of the user interaction data, apart from a task-related question the users have come to our bot with the following intents:

- **Perform a task unsupported by the Taskbot**: For Example- playing music or performing general functions that Alexa can do.
- **Have a General Conversation**: For Example- trying to chit-chat and have a conversation with the bot, is more suitable for a social bot, etc.
- **Need help with what to cook**: For Example- deciding on what to cook, needing recommendations for recipes, etc.
- **Need help with what task to do**: For Example- deciding on what DIY task to do, needing recommendations based on the season, etc.
- **Talk about different entities**: For Example- talking about their pets, different animals, etc.
- **Express their feelings**: For Example- talking about how they are feeling, needing emotional comfort, etc.

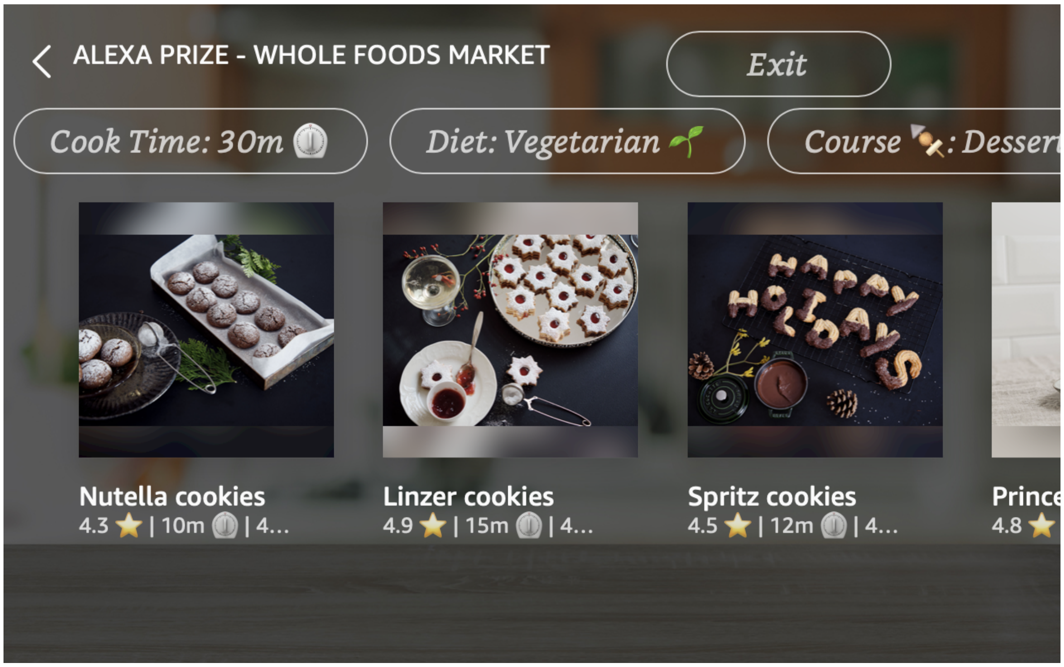

Figure 10: The visual interface for our Whole Foods query results.

K Interface Design

K.1 Query Results Page

A screenshot of our query results page can be found in Figure 10. This page is displayed when the user has given a search query and ISABEL has found valid search results to display to the user as a list of options. The user can scroll through the options, and clicking on a recipe will give them more details about that recipe. We provide a dynamic, scrollable list of filter buttons directly above the search results, which users can click to filter their results based on specific attributes. We also use emojis to give the users a short, eye-catching summary of some important details of the recipe, including the rating, number of steps, and required cooking time.

K.2 Task Detail pages

Task Overview Page Upon clicking a task, the user may need more information about the task before they view the steps. For recipe tasks, the user must know the task ingredients. For tasks with multiple methods and parts (which each contain multiple steps), the user may only wish to browse the steps for a particular method or part. As such, we provide a task overview page that is displayed when the user starts the task, where basic information about the task is displayed and users can view the ingredients and navigate to the steps (recipes) or view a task-specific video and choose the method they would like to start with (how-to tasks) Screenshots of the overview pages for recipe and how-to tasks can be found in Figure 12.

Scrollable Steps Page Accommodating large amounts of information in an easily digestible and navigable format is vital for accessibility and user-friendliness. To achieve this, we've implemented 'Scrollable Features' in our system. Areas such as ingredients, materials, methods/parts, and steps, which may involve extensive lists, are scrollable. Such scrollable lists can be found in the task overview pages (Figure 12) as well as in separate scrollable lists of steps (Figure 12).

Detail Page The 'Detail Page' is one of the critical components, offering users an expanded view of each task step. From the 'All Steps' page, users can select an individual step, which then opens the corresponding Detail Page. Here, they can access in-depth instructions and additional information

Figure 11: The task overview pages for recipes (left) and how-to tasks (right).

about the step (when provided in the recipe metadata) including an image depicting the step and the ingredients needed for that step. Figure 13 contains a screenshot of a detail page for a recipe task.

Figure 12: The scrollable list of recipe steps.

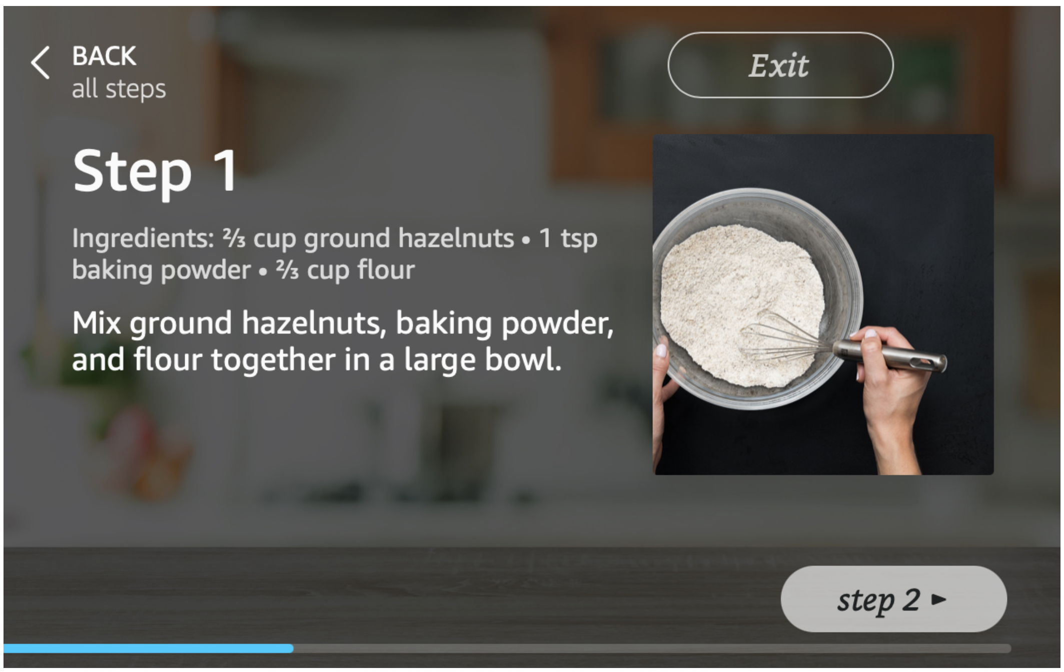

Figure 13: The detail page for recipe steps.