TWIZ: A Conversational Task Wizard with Multimodal Curiosity-Exploration

Rafael Ferreira, Diogo Tavares, Diogo Silva, Frederico Vicente, Mariana Bonito,
Gustavo Gonçalves, Rui Margarido, Paula Figueiredo, Helder Rodrigues,
David Semedo, Joao Magalhaes

Abstract

This paper describes the vision, scientific contributions, and technical details of the Task Wizard (TWIZ) team's participation in the Alexa TaskBot Challenge 2021. Our bot design envisions the support of an engaging experience, where users are guided through multimodal conversations, towards the successful completion of the selected task. This is achieved through four key principles: a) robust dialog interaction, by making core dialog components (utterance processing, dialog manager, and response generator) tailored for an in-the-wild setting, b) effective and conversational task grounding, c) delivery of an immersive multimodal task guiding experience, and d) engagement maximization and user cognitive stimuli. In the following sections, we present the proposed methods to tackle each one of these principles, leading to a novel multimodal curiosity-exploration task guiding conversational assistant, that manages to balance user engagement and cognitive load, while guiding users through complex tasks.

1 Introduction

Conversational agents have emerged in our daily lives by helping us in a multitude of tasks. Despite its success, specific tasks such as cooking and Do It Yourself (DIY), task complexity starts to become an issue that quickly leads to user frustration. We propose a multimodal task-guiding agent, that by leveraging on a robust dialog framework, seeks to guide users in a smooth and engaging manner. The core idea is to distribute user's cognitive load throughout the task stages, while keeping it fun and engaging. The multimodality facet is exploited to provide visual depictions that complement the agent's utterances. This symbiosis between the task completion goal and the multimodal curiosity-exploration paradigm is at the heart of the Task Wizard (TWIZ) bot.

In this context, the key contributions of our bot are as follows:

- **Task organization and presentation.** To improve the clarity of complex tasks, we propose to decompose the tasks into simpler steps that are then automatically illustrated with images. To better guide users through complex steps, we also introduce supporting step specific videos, when available.
- **Conversation task-grounding.** To help the user quickly find the task and minimize frustration, we implemented a robust intent detection; a Transformer-based ranker; and seasonal suggestions for the users that only wished to explore the TaskBot.
- **Conversational engagement.** To keep the user engaged throughout the conversation, we provide a 3D visual preview of the task; introduced task specific curiosities through dialog system initiative; and support domain specific question-answering in conversation.

1st Proceedings of Alexa Prize TaskBot (Alexa Prize 2021).

2 Architecture

Our approach is presented in Fig. 1. It is based on the CoBot framework provided by Amazon for developing conversational agents. This framework was designed to run the conversational agents on cloud functions (AWS Lambdas) which take advantage of server-less computing benefits. We take advantage of the framework's layered architecture and use it with a shared database pattern. We opted to run the Machine Learning algorithms on docker containers in remote computing platforms (known as remote modules). This was done to reduce the load on the Lambda running the conversational agent. Furthermore, we create dedicated dialog state-aware Response Generators to treat each user intent and conversational functionality as detailed in Sections 3 and 4.

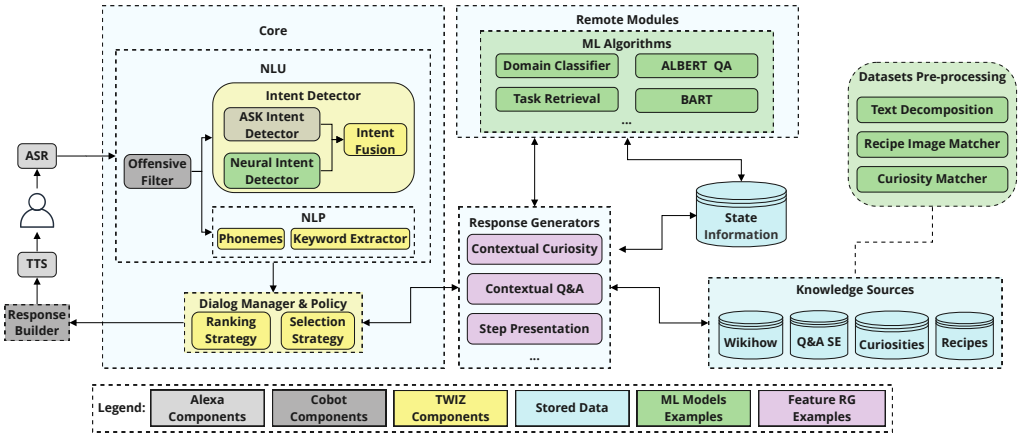


Figure 1: TWIZ's components architecture.

3 Robust Dialog Interaction

This section describes the components required to provide a robust dialog interaction. Our goal was to establish a main task flow, provide flexibility to support out of scope interactions, and gently redirect the users towards the desired task flow. We start by defining the stages of the Dialog Manager, which establishes the main interaction with the bot. Secondly, we present our Intent Detection framework to support the users' requests. Next, we focus on diversifying the generated dialogues. Finally, we present our enhanced support for Task Search.

3.1 Dialog Manager

The dialog manager is responsible for keeping the flow of the conversation, and allows the user to interact with the various components of TWIZ. We designed our interaction to consider the device the user is using, since a multimodal interaction is inherently different from a voice-only experience. We identified 6 major stages in the expected dialog flow, listed below:

1. **Start** - This stage includes the greeting, followed by an explanation of the allowed requests, along with specific suggestions to increase user engagement, Section 4.1.1.

2. **Task Grounding/Search** - This stage is achieved after the user issues a valid search request as classified by our domain classification module, Section 3.5.1, and presents the results retrieved by our search pipeline, Sections 3.5.2 and 3.5.3. We carefully considered how to present the results to the user depending on the type of device, showing fewer results on screenless devices to reduce the user's cognitive load.

3. **Task Overview** - After the user selects one of the provided tasks. We show a brief task overview (e.g., number of parts, duration, difficulty), allowing the user to confirm that the task is in the user's expected parameters.

4. **Step-by-Step** - At this moment the user begins the task and the bot starts to read the steps. We put significant effort into modeling each step by decomposing a large step into various smaller ones, Section 4.2.1, to better guide the user through the task. In multimodal devices, for recipes that do not have images associated, we also match the step text to the image of another recipe to take advantage of the multimodality aspect, Section 4.2.3. In this stage, the user can also ask questions, Section 4.3, and to increase user engagement, we prompt the user with contextual curiosities, Section 4.3.3.

5. **Steps List** - This stage is only available for multimodal devices, since it is used for the user to quickly view and navigate through all the steps.

6. **Task Concluded** - This is the last stage of the dialog that happens when the user reaches the end of the task. We ask for confirmation to leave the bot, and thank the user before ending the interaction with a custom message depending on the task category along with a contextual curiosity.

In addition to the previous stages, the dialog manager also allows contextual support ranging from contextual help prompts, to specific fallback responses for common use cases such as asking to play music, or open another skill. All of these prompts are very useful in guiding and educating the user to interact with the bot.

3.2 Intent Detection

A robust intent classification module is required to bolster both the transitions between the dialog stages, and the successful detection of user requests within said stages. We define 37 intents to support a large variety of distinct interactions. Intent detection is tackled in two distinct steps: First, we attempt to classify it using an ensemble of methods. Then, we perform minute rule-based corrections.

3.2.1 Intent Detection Pipeline

The performance of the intent detection module was maximized by focusing on making our system robust to ASR errors and generally noisy user utterances. This is achieved by applying phonemes-based matching and neural intent classification, respectively.

Phonemes-based matching: We compare the phonetic spelling [15] of a set of manually curated user utterances and the input utterance, to improve robustness to ASR errors. The similarity score is calculated between both sequences, and the intent is accepted if its tokenized similarity score is over a certain threshold. This strategy is only employed for a subset of all intents to reduce false negatives in intents with high variability, such as *Questions* and *Search queries*.

BERT-based neural intent detection: We manually collect and annotate an intent detection dataset to facilitate the evaluation of the TaskBot. The dataset is composed of 1014 real dialogs and 7994 user-system utterance pairs—spanning every dialog without empty user utterances from the start of the competition until early March—using our set of possible intents as reference. Furthermore, for *Search queries* and *Task selection* intents, we annotate the slots which are referenced by the user. The data is inherently unbalanced, as it directly reflects the interactions real users had with TWIZ – for example in typical dialogs, *Next* intents will be more frequent than *Question* intents.

Evaluation results are shown in Table 1, with BERT referring to the original model [3] with an intent classification layer. Appended to the model name, we describe the input information as follows: U signaling the usage of a user utterance, A an agent utterance, and $stage$ a token representing the conversation stage indicated in Section 3.1. Note that in the final evaluation step, we don't consider the most common intents with the lowest variability (*Next, Yes, No*), to highlight the disparity between approaches, since accuracy for the lowest variability intents is close to 100%.

Table 1: Neural Intent Detection Results.

Model	**Intent detection acc.**
BERT (U)	81.6%
BERT (U+A)	82.5%
BERT (U+stage)	83.4%
BERT (U+A+stage)	83.6%
BERT-large (U+A+stage)	**83.8%**

Intent fusion is performed by comparing the probability outputs of the BERT intent classification layer, the results of the phonemes score, and Alexa ASK console. We use the BERT output in two scenarios: if Alexa ASK console outputs a *Fallback* intent, or when BERT's confidence is high (≥ 0.9). If the phonetic matching results in a sufficiently high similarity (≥ 0.75), we choose its output instead. The thresholds were empirically set by replaying user conversations.

3.2.2 Rule-based Corrections

A rule-based final step is used to correct the decision of the previous module, in unseen scenarios (zero-shot). Rules are derived from potential candidate utterances that are likely to occur in live user conversations, to guarantee that specific features are activated. For each rule, we create dedicated regular expressions that match them. Throughout the competition, we monitored user conversations and updated our rules with the most common utterances. This way, new intents can be swiftly added to the system if the need arises. Training data can then be collected from user interactions, and used to re-train the neural module.

3.3 Supporting Natural and Human-like Task Selection

While our Intent Detector covers our standard commands, and their variations, it can struggle with user utterances that pertain to the displayed content, such as tasks. This poses a challenge for the user to select a task by referring to them by name as they would in a human-like interaction. To achieve this, we computed smooth *TF-IDF* [6] vectors, for the query and each result, and then used the cosine similarity to score each query-result pair. However, this can lead to inaccurate matches if only a small part of the query terms is contained in the results, e.g., "vegan lasagna" would match with "lasagna" if vegan wasn't a known term. To mitigate this, we expand the vocabulary with a curated set of search modifiers, such as "roasted" or "vegan", and we also apply a penalty component to the cosine similarity score, based on the length of the query and its intersection with the vocabulary. With this, the similarity score between query q and result r is:

$$score(q, r) = \cos_s(q, r) * (-\alpha^{(\frac{q \bigcap vocab}{\|q\|} - 1)} * \beta + \gamma), \tag{1}$$

where α, β, and γ are parameters tuned with values 20, 0.9, and 1.045, respectively. This meant that off-topic queries and potential new searches would be recognized as such. To reduce the amount of false negatives, we also removed stopwords and stemmed both the query and results.

3.4 Answer Templates & Generation

The nature of the TaskBot and the previously mentioned conversation stages, introduces a finite number of conversation states. For most cases, we follow a template-based answer realization approach, as it allows us to design a seamless interaction with little risk of off-topic or incorrect answer generation. The following key aspects were considered while designing answers:

- **Utterance Length** is a critical design choice to improve the user comprehension of the bot's responses. We sought to keep utterances short, while keeping each utterance focused on critical information;
- **Utterance Tone** consistency is important to maintain a comfortable user experience. We focused on a casual, but polite tone, making use of Speech Synthesis Markup Language (SSML) to introduce emotion and prosody. This allowed us to make the utterances more natural and engaging;
- **Utterance Diversity** is an imperative component of an engaging conversational agent. As users traverse the same conversation stage multiple times, it becomes crucial to reduce response repetitiveness. We address utterance diversity by producing several response variations, and pseudo-randomly selecting them to avoid repetitions.

3.5 Task Search

An effective search component is paramount for system performance to ground the conversation on the correct domain and provide users with the most relevant tasks.

3.5.1 Domain Classification

User requests need to be classified into a domain to ground the conversation. Three domains were considered: recipes, DIY, and out-of-scope. Having no amount of human-annotated data at scale for this particular task, we used the WikiHow Dataset [25] as a source of training data, and consider WikiHow articles' categories as labels. First, we extracted all categories, and manually classified the 50 most common categories into one of the three domains. In a second step, we add a new "Dangerous" domain by reclassifying potentially dangerous tasks, e.g., "how to drive a bulldozer", using a rule-based approach against a manually-built dictionary.

Given the positive performance of Transformer-based [22] models in text classification tasks [3, 20, 11], we used BERT [3] and its distilled version [20] to classify the request's domain, by applying a linear layer on top of the $[CLS]$ token. The input of the model is the title of the article with a random chance of adding a prefix or suffix to simulate a user query (e.g. "*can you help me with {article_title}*"), and the output is a distribution over the four categories.

We validate our approach against the selected WikiHow Articles, and to further assess model generalization, we validate it against 285 manually annotated user queries. Table 2 shows the domains' distribution across datasets and the results of each of the models. As we can see, both models behave similarly and the results are not particularly affected by model size, as seen by the performance of the distilled model.

Table 2: Dataset Label Distribution and Domain Classification Model Results.

Domain	**WikiHow Articles**	**Annotated Queries**
DIY	40914	143
Recipes	12956	99
Out-of-scope	12641	31
Dangerous	11595	12
Model	**Accuracy**	
DistilBERT	0.885	**0.888**
BERT BASE	**0.891**	0.867

3.5.2 Text-based Retrieval

The first step in our search pipeline is to process the query text in order to focus on the most important terms. Our first approach to address this requirement uses Spacy [5] to perform part-of-speech tagging (POS) and dependency parsing. With the words tagged, we remove stopwords and focus on verbs, nouns, and noun-phrases. Upon analysis of the search results, we concluded that DIY task search improved by considering both verbs and nouns. However, recipe results improved by only taking into account nouns and some cooking-related verbs (e.g., grill and fry). This allows us to effectively search in the provided Amazon's recipe and WikiHow APIs by removing irrelevant terms, greatly increasing the performance of the search pipeline.

3.5.3 Dense Passage Retrieval

While the provided APIs allow for text-based search of tasks, we aspired for a search component that was more robust to noisy queries. As such, we utilized the Dense Passage Retrieval model, DPR [7]. This model uses a dual encoder (BERT-based) architecture and allows for semantic search on the available articles. For in-domain training, we created a synthetic dataset, which was generated using the T5 [17] query generator[1]. This model, based on the T5 model [17], generates reading comprehension-style questions with answers extracted from a text. In this case, using tasks from both the WikiHow dataset [25] and tasks previously retrieved from the APIs, we generate relevant query-document pairs. This training procedure can be found in Fig. 2. Here, we use articles and their respective questions generated by T5 as positive samples, and sample distinct random Article-generated question pairs to be used as negative samples.

Additionally, we manually annotated the relevance of tasks retrieved from the provided APIs for queries issued by our users. This allowed us to use relevant query-document pairs as positive samples

[1] `https://huggingface.co/iarfmoose/t5-base-question-generator`

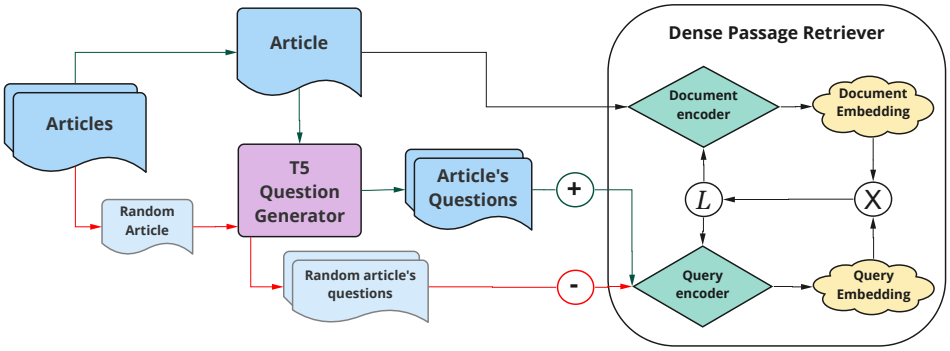


Figure 2: DPR training on synthetic dataset.

for DPR, while using irrelevant pairs as negative samples, to further fine-tune the model. Finally, when a user issues a query, we parallelize the retrieval process by searching both on the API and DPR, and fusing the ranking results according to the Jaccard similarity against the query.

4 Multimodal Conversational Engagement

One of our central motivations was to guarantee an enjoyable interaction with TWIZ, and improve the overall user experience through the creation of well-paced and engaging interaction flows. Hence, to mitigate the risk of creating monotonous assistants, we investigated a series of strategies to step-up user engagement by enriching the multimodal dimension of conversations.

4.1 Task Grounding Stage

The task grounding stage of the TaskBot is critical to ensure that the user is able to search and start a task. Users try multiple searches, many times completely unrelated, essentially exploring the TaskBot capabilities. Moreover, asking too many questions and making open statements can push the user towards ambiguous requests. This behavior turned out to be both a challenge and also an opportunity which we tackled using both contextual suggestions and task highlights.

4.1.1 Contextual Suggestions and Task Highlights

A landing page is the root element of modern UX interactions [13], where users fallback when they get lost and are discovering interaction flows. Providing an intuitive initial screen is imperative for new users in our TaskBot. At this stage, we use suggestions which provide an exploratory learning experience while the user still hasn't discovered the bot's capabilities.

While displaying the top results, or suggestions, task highlights are used to persuade the user to promptly start a task. We create different prompts for various task dimensions such as difficulty, number of ratings, and duration. The system will provide an appealing prompt about one of the task highlights, instead of always reciting all of the task dimensions. Task highlights are implemented across voice-only and multimodal devices, to increase the dialog dynamic and provide conversational snippets, or previews, about tasks before selecting them.

4.1.2 3D Visual Previews

Multimodal devices present opportunities for richer interactions. Users are familiarized with a desktop paradigm [13], and voice-based interactions present new challenges, even in smaller devices with limited screen sizes. The TaskBot Challenge presents exciting opportunities and challenges while exploring this shift towards voice-based interactions, in an unforgiving real-user feedback scenario.

Multimodal devices have a clear edge to captivate the users' attention and are able to persuade users to interact with a task. TWIZ presents a video with a 3D Ken Burns effect [14], which can be observed in Fig. 3. The video zooms out and pans from a tiramisu cake, where we can notice the increasing distance between the saucer, the bottom left part of the tray, and the upper corner of the tiramisu cake.

As the zoom progresses from left to right in Fig. 3 the distance between these elements increases, thus suggesting a 3D illusion.

This effect is displayed during the recipe overview stage, where the visual effect is generated from the recipe thumbnail image. We use this effect to promote the swift selection of a task, while fully using the available screen size.

Figure 3: Example of a 3D Video Ken Burns Filter where the distance between objects increases as the perspective changes: Video Available Here.

4.1.3 Search Fallbacks

A common pattern that we observed was the repetition of the same query when the search results did not meet the user's request. This was a potential cause for frustration and loss of engagement. To mitigate these situations we designed a different conversation flow to acknowledge the conversation repetition and propose a task suggestion to the user. As future work, we plan to investigate query performance prediction methods.

4.2 Enhancing Step Presentation

To ensure a smooth path between task steps, it is important to manage users' cognitive load, by breaking complex steps into simpler ones. TWIZ also complements the step textual descriptions with visual elements. Steps are accompanied by videos, or images, which provide a multimodal dimension that further clarifies the step's instructions.

4.2.1 Task Decomposition

One key aspect of a TaskBot is the ability to guide the user in a well-paced manner. The original tasks are decomposed into steps, however, these were not created with a conversational assistant in mind. Extensive steps will overwhelm the users with information, which is especially prominent in screenless devices as the users rely on their short-term memory [2, 12].

We created methods capable of decomposing each step into manageable pieces of information, as represented in Fig. 4, to mitigate the information overload. Here, a single step was decomposed into two logical steps, and the resulting steps are easier to follow in a task guiding setting for both device types.

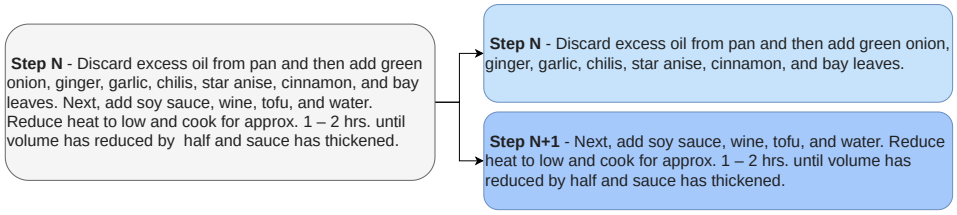


Figure 4: Example of the decomposition of a complex step.

Two task decomposition algorithms were investigated. The first method is an unsupervised approach based on step length. We use Spacy [5] to detect sentence boundaries, and break the step after a certain word threshold has been reached. The second method, *Task-Token-Segmenter*, is a supervised method which uses a BERT [3] model to learn the best places to break a step. We use the embedding

of each token as input to a shared linear layer that does a binary prediction (*segment/not segment*). This way, we are able to segment a step based on the length while also considering its content by learning specialized embeddings. We considered a subset of recipes and removed recipes shorter than 3 steps, and with steps longer than 100 words (i.e. too long). Using these well-formatted recipes and their corresponding steps, we follow a similar line of work in the Wikipedia domain [8, 1], and use the original step segmentation as the ground-truth labels to train the model. This trained model was then used to decompose long recipe steps.

4.2.2 Step Specific Videos

Video illustrations provide an explicit visual explanation of the step's text, thus elucidating possible doubts of following complex steps with only textual instructions. When available, we introduce step videos to both recipes and Wikihows on the right side of the screen. We pre-processed the step videos extensively generating an overlay step video over the thumbnail image as background. The videos quality was ensured by processing for the different resolutions of Echo Show 5, 8, and 10 devices. This effort resulted in many hours of video processing, producing a high-quality multimodal experience. Moreover, we allow the user to zoom-in on the respective step video, thus taking full advantage of the available screen size, where the user can listen to the step instructions while observing attentively the video details.

4.2.3 Automated Recipe Steps Illustration

Step-by-step instructions can be accompanied by a step image, which illustrates a portion of the recipe instruction. While using the provided APIs, we noticed that a great majority of recipes did not provide step-by-step images. As a solution, we gathered similar images from recipes that share similar steps text and included them in steps that originally did not have illustrations.

To achieve this, we first perform text similarity between the non-illustrated recipe and the set of illustrated recipes. In particular, we use the recipe's details, namely title, ingredients, and steps, and apply a weighted *TF-IDF* [6] to obtain the top-n candidate steps with corresponding images. After having this first list, we apply the pre-trained multimodal model, CLIP [16], which takes as input the recipe's text and the candidate image, and yields a similarity score between the two. Finally, we rank the results by similarity. We experimented with an alternative method for text matching using BLINK [23]. We extracted all the entities and then we calculate the similarity using *TF-IDF*, where instead of word coexistence, we consider named entities coexistence, weighted by the BLINK score.

To evaluate our approaches, we manually annotated the results of each model on a 0 to 2 scale, where 0 means non-relevant, 1 means relevant (the image depicts some elements of the step) and 2 means a highly relevant image illustration. We introduced two kinds of control annotations for an exact image and a wrong image and converted the 0 to 2 annotations to a weighted 0 to 100 score. The results are shown in Table 3. Our best method (Top 5 TF-IDF > Top 1 CLIP) selects a good illustration in more than 70% of the cases.

Table 3: Automated Recipe Steps Illustration Results.

Method	**Score**	**#Annotations**
Control: Original Image	85,5	104
Top 5 TF-IDF > Top 1 CLIP	71,0	92
Top 10 TF-IDF > Top 1 CLIP	70,9	87
Top 20 TF-IDF > Top 1 CLIP	69,4	90
Top 20 BLINK > Top 1 CLIP	65,5	90
Top 5 BLINK > Top 1 CLIP	63,4	91
Top 10 BLINK > Top 1 CLIP	60,5	91
Top 1 TF-IDF	55,6	90
Top 1 BLINK	49,6	87
Control: Random Image	10,3	139

4.3 Task Execution Stage

Executing a task can be a daunting process. We observed that most users did not execute the task until the end and would quit halfway through. To minimize these withdrawals, we resorted to ways of making the task more engaging. Hence, our bot can take the dialogue initiative and propose to tell users curiosities about the recipe. Moreover, we enriched our assistance by supporting questions about the discourse of a task, with contextual questions, external knowledge bases, and by introducing ingredient and tool substitutions.

4.3.1 Contextual QA

Mid-task user queries need to be identified and addressed. For this, we resort to a large version of ALBERT [9] fine-tuned on SQuAD2 [18], which we gradually feed with augmenting task knowledge depending on whether the current context has fulfilled a user question. The starting context is the question and the current task step, which is further extended to all other steps if no answer is found. If still no valid answer span is found, we use the ingredient list or the tools list as context.

Another common request when executing a task is the lack of a particular ingredient or tool. We use a dataset of replacements [21], to complement Amazon's EVI API.

4.3.2 BART: Talk Human to Me

We wanted TWIZ to personify a friendly assistant, so users could trust and be enlightened by a more natural interaction. Regarding the end-goal and domain of this competition, we were bounded in guiding users through the execution of recipes and DIY tasks. In turn, combining both human-like and a strictly bounded interaction naturally becomes a challenge when attempting more creative interaction flows, while ensuring an expected flow behind tasks' assistance. Having this in mind, we weighted both the hallucination risks and human-like answering possibilities and put together a text generation BART [10] model for two different types of purposes. One was related to answering general questions in a more verbose sense, specifically concerning the domains at hand. The second purpose consisted in the ability of elaborating on curiosities, which occur from time to time. For instance, prompting a question or an elaboration request (e.g. "why?", "tell me more about that") after a curiosity, TWIZ attempts to reason about it and provide a friendly response to the user (See Fig. 5 for an example).

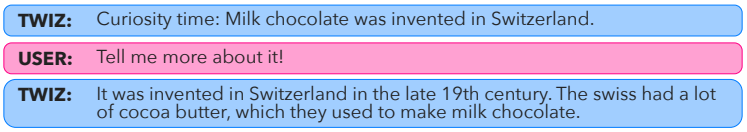


Figure 5: BART elaboration on a curiosity.

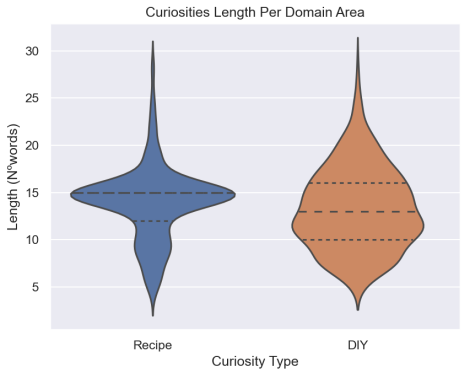
To fine-tune the BART model, we collected several different datasets and pre-processed them to remove non-linguistic textual artifacts (e.g. URLs and image references). The data collected was the following: the eli5 and the askScience subreddit dump [4], and additionally, we captured questions and answers from 5 communities (cooking, crafts, DIY, gardening, and pets) of the stack exchange forum[2]. This variety of QAs textual data permitted BART to gain insights over the domains of this challenge.

Our fine-tuned model also had a surprising tendency to contradict itself, which we tried to abolish by detecting negation of statements.

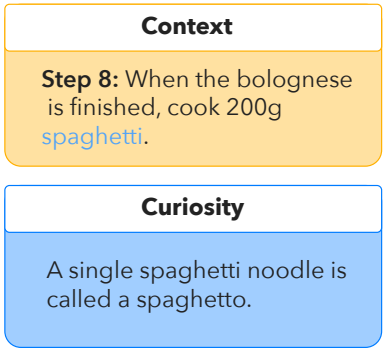
4.3.3 Curiosities

The concept of **curiosity** has been attempted to be computationally modeled in [24], by considering measurable features such as complexity or knowledge conflict. TWIZ acknowledges a curiosity as being: a natural language phrase which ignites a reader to seek for more information, while providing an extension to the reader's knowledge-rich in athirst data (See Fig. 6 for a curiosity example). Taking

[2] `https://archive.org/download/stackexchange`



(a) Length distribution of the Curiosities dataset



(b) Curiosity phrase example, in context

Figure 6: Curiosity insights.

this into consideration and with the vision of enriching TWIZ's interactions with contextualized curiosities throughout the task execution, we created the Curiosities Dataset. Currently, the dataset consists of 1350 curiosities distributed through both the recipes and DIY domains with 743 and 607 curiosities, respectively. To ease the introduction of curiosities, we made sure to lower their complexity by using short and objective sentences, averaging around 15 words (See Fig. 6).

Enriching Tasks with Curiosities: Curiosities are introduced to boost engagement during the execution of a task. These curiosities are contextualized on the ongoing task to ensure relevancy. This contextualized correspondence is performed through a two-step process, using Sentence Transformers [19]. First, curiosities and the task's text characteristics (e.g. title and steps) are encoded through a Bi-Encoder model. Next, a cosine-similarity score is attributed to each pair from which we choose the top-10 curiosities. As an attempt to remove some false-positives, we have a second step, which consists of passing the top-10 results from each task's characteristics through a Cross-Encoder [19] to re-rank them and choose the top-3, that best relate to the context at hand.

5 Automatic Tests

As TWIZ's complexity increased, testing its correct behavior becomes exponentially harder. Since the TaskBot's rating is deeply tied to user satisfaction and competition progression, we must ensure that we do not introduce unwanted behaviors when new functionalities are added. As such, we built upon the provided Cobot's interactive testing functionality, allowing our team to create repeatable interaction flows easily and fast.

We designed a comprehensive set of dialog flows, covering not only the certification guidelines but also potential conversation paths that activate specific TWIZ features. The creation of these flows was done manually only once and allowed for significant time savings, thus greatly reducing manual labor by passing through all aspects of the bot where the behavior was left unaltered.

6 Conclusions

This paper summarized TWIZ's work for the first edition of the Alexa TaskBot Challenge. Constant analysis of the user conversations and feedback has helped us understanding what issues needed more attention. Our initial concern was to build a robust and solid dialog interaction foundation, to work in an *in-the-wild* setting. We then exploited the multimodal capabilities of Alexa devices and promoted engagement in a way that accounts for the user's cognitive load. Feedback has also informed us about which aspects were more valued by users. One of the best-received features was the curiosity facts, which brought increased engagement, and the usage of videos, which is a major help in situations where steps are complex.

References

[1] ARNOLD, S., SCHNEIDER, R., CUDRÉ-MAUROUX, P., GERS, F. A., AND LÖSER, A. SECTOR: A neural model for coherent topic segmentation and classification. *Trans. Assoc. Comput. Linguistics 7* (2019), 169–184.

[2] COWAN, N. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and brain sciences 24*, 1 (2001), 87–114.

[3] DEVLIN, J., CHANG, M., LEE, K., AND TOUTANOVA, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT* (2019), Association for Computational Linguistics.

[4] FAN, A., JERNITE, Y., PEREZ, E., GRANGIER, D., WESTON, J., AND AULI, M. ELI5: long form question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers* (2019), A. Korhonen, D. R. Traum, and L. Màrquez, Eds., Association for Computational Linguistics, pp. 3558–3567.

[5] HONNIBAL, M., MONTANI, I., VAN LANDEGHEM, S., AND BOYD, A. spaCy: Industrial-strength Natural Language Processing in Python.

[6] JONES, K. S. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation 60*, 5 (2004), 493–502.

[7] KARPUKHIN, V., OĞUZ, B., MIN, S., LEWIS, P., WU, L. Y., EDUNOV, S., CHEN, D., AND TAU YIH, W. Dense passage retrieval for open-domain question answering. *ArXiv abs/2004.04906* (2020).

[8] KOSHOREK, O., COHEN, A., MOR, N., ROTMAN, M., AND BERANT, J. Text segmentation as a supervised learning task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)* (2018), M. A. Walker, H. Ji, and A. Stent, Eds., Association for Computational Linguistics, pp. 469–473.

[9] LAN, Z., CHEN, M., GOODMAN, S., GIMPEL, K., SHARMA, P., AND SORICUT, R. Albert: A lite bert for self-supervised learning of language representations, 2019.

[10] LEWIS, M., LIU, Y., GOYAL, N., GHAZVININEJAD, M., MOHAMED, A., LEVY, O., STOYANOV, V., AND ZETTLEMOYER, L. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 7871–7880.

[11] LIU, Y., OTT, M., GOYAL, N., DU, J., JOSHI, M., CHEN, D., LEVY, O., LEWIS, M., ZETTLEMOYER, L., AND STOYANOV, V. Roberta: A robustly optimized BERT pretraining approach. *CoRR abs/1907.11692* (2019).

[12] MILLER, G. A. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review 63*, 2 (1956), 81.

[13] NIELSEN, J., AND TAHIR, M. *Homepage Usability: 50 Websites Deconstructed*. 2002.

[14] NIKLAUS, S., MAI, L., YANG, J., AND LIU, F. 3d ken burns effect from a single image. *ACM Transactions on Graphics 38*, 6 (2019), 184:1–184:15.

[15] PARK, KYUBYONG & KIM, J. g2pe. `https://github.com/Kyubyong/g2p`, 2019.

[16] RADFORD, A., KIM, J. W., HALLACY, C., RAMESH, A., GOH, G., AGARWAL, S., SASTRY, G., ASKELL, A., MISHKIN, P., CLARK, J., KRUEGER, G., AND SUTSKEVER, I. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning* (18–24 Jul 2021), M. Meila and T. Zhang, Eds., vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 8748–8763.

[17] RAFFEL, C., SHAZEER, N. M., ROBERTS, A., LEE, K., NARANG, S., MATENA, M., ZHOU, Y., LI, W., AND LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv abs/1910.10683* (2020).

[18] RAJPURKAR, P., JIA, R., AND LIANG, P. Know what you don't know: Unanswerable questions for squad, 2018.

[19] REIMERS, N., AND GUREVYCH, I. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[20] SANH, V., DEBUT, L., CHAUMOND, J., AND WOLF, T. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR abs/1910.01108* (2019).

[21] SHIRAI, S., SENEVIRATNE, O., GORDON, M., CHEN, C., AND MCGUINNESS, D. L. Semantics-driven ingredient substitution in the foodkg. In *Proceedings of the ISWC 2020 Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 19th International Semantic Web Conference (ISWC 2020), Globally online, November 1-6, 2020 (UTC)* (2020), K. L. Taylor, R. S. Gonçalves, F. Lécué, and J. Yan, Eds., vol. 2721 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 242–247.

[22] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, L., AND POLOSUKHIN, I. Attention is all you need. In *NeurIPS* (2017).

[23] WU, L., PETRONI, F., JOSIFOSKI, M., RIEDEL, S., AND ZETTLEMOYER, L. Zero-shot entity linking with dense entity retrieval. In *EMNLP* (2020).

[24] WU, Q., MIAO, C., AND SHEN, Z. A curious learning companion in virtual learning environment. pp. 1–8.

[25] ZHANG, L., LYU, Q., AND CALLISON-BURCH, C. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 4630–4639.