UMass Proposal to the Alexa Prize TaskBot Challenge 2022

Abstract

UMass participated in the Alexa Prize TaskBot Challenge 2021. Based on the rich knowledge and experience gained during last year's participation, in this proposal, we describe our proposed system for the second iteration of the challenge. We plan to develop three pipelines: (1) offline task instruction mining and enrichment, which prepares the data required for an effective TaskBot, (2) the conversational agent, which consists of several components from intent identification to retrieval to question answering, and (3) monitoring and self-learning, which prepares a human-in-the-loop infrastructure for error detection and analysis and iterative improvement of the TaskBot. We will emphasize on multi-modal enrichment of the conversations by a mixture of image retrieval, image generation (using adapting the recent text-to-image models), and short video retrieval. Based on our past experience, we believe that we can not only develop a competitive bot, but also conduct cutting-edge research in multi-modal conversational AI.

Our Proposed System: MarunaBotV2

Depending on our extensive experience in designing and developing task-oriented dialogue systems, we will design and develop MarunaBotV2 according to the following two core principles: *robust performance* and *customer-focused design*.

Robust Performance: In the last participation of UMass at TaskBot Challenge, we learned that a predefined conversation flow can only perform well in "perfect" scenarios where the user's behavior is predictable. However, we observed that users often behave irrationally. Therefore, in designing the MarunaBotV2 architecture, we will carefully consider *robustness* as a core principle. Meaning that if a user changes their mind in the middle of a task, asks unrelated questions, requires chit-chat conversations, and so on, MarunaBotV2 will be able to respond appropriately. In addition to considering robustness in our model design choices, we will develop a system with a unified confidence score prediction for any (rational or irrational) interaction and MarunaBotV2 will ask clarifying questions when needed to avoid significant failures.

Customer-Focused Design: We know that robustness and advanced response generation capabilities will lead to a good conversation, but a careful customer-focused design will make a great system that customers will love to use. This type of design involves many aspects, including responsiveness (using efficient models), careful dialog design with a variety of responses, and mixed-initiated interactions to ask for clarifications or make suggestions, and large-scale monitoring of satisfaction-related metrics. One of our team members will be more focused on this aspect by analyzing weekly interactions we receive from users, making sure that all our design choices are leading to user satisfaction.

Based on the mentioned principles, MarunaBotV2 will be composed of three pipelines: (1) task instruction mining and enrichment, (2) conversational agent, and (3) monitoring & self-learning. Task mining enables us to collect required data that can be later consumed by the conversational agent. Monitoring and self-learning will be a mixture of manual and automated processes for measuring satisfaction-related metrics for past conversations.

Pipeline 1: Task Instruction Mining and Multimodal Enrichment

"Data is the new oil!" From our last participation, we learned that WholeFoods and WikiHow APIs are limited and do not provide all the information necessary for making robust interactions for all kinds of user information needs. In addition, we learned that solely accessing the APIs without learning precise representations of task instructions for retrieval (e.g., using state-of-the-art dense retrieval models) is a bottleneck in retrieval performance. Therefore, we propose to mine information, such as task instructions, from multiple sources. For task instruction mining, we will explore two different approaches: (1) web-based mining of tasks using a web search API, and (2) offline construction of a task corpus using CommonCrawl. In both cases, the pipeline will have three steps: (1) web page classification, (2) instruction steps extraction and (3) steps enrichment with relevant visual components. Step 1 will identify which web pages are most likely about instructions for a task/goal based on the title. The second step will employ a weakly supervised transformer model to identify the various steps. We will explore extractive and/or abstractive language model-based methods for this task. Given the focus of TaskBot Challenge, we believe enrichment with visual content is important. For the enrichment of instructions with images in step 3, we will first mine a set of objects and actions (verb-object tuples) from the content of task instructions. Then we will explore two methods for visual enrichment: (1) CLIP-based search on a corpus of crawled images and (2) offline generation of relevant images using a text-to-image generative model (e.g. Stable Diffusion [1]).

Pipeline 2: Conversational Agent

This is the main pipeline of our system and implements the conversational agent that retrieves relevant task instructions, guides the user through the steps of the selected instruction, answers questions, and discusses and executes commands. This pipeline is depicted in Figure 2 in details. This pipeline is composed of 7 types of neural models; intent classification, retrieval & re-ranking, slot filling, option selection, question answering, question/answer re-writing and chit-chat response generation.

Intent Classification: Based on our recent research on few-shot classification, we will develop a robust intent classifier with self-supervised pre-training on millions of dialogs from Reddit, few-shot contrastive fine-tuning on examples from the recent Wizard of Tasks dataset, and additional synthetic or curated examples through crowdsourcing. For synthetic data, we will explore the use of prompting in large-scale pre-trained language models (OPT, Bloom, Flan-T5).

Retrieval Models: Task instruction retrieval is a crucial part of a TaskBot. For retrieving recipes & DIY task instructions, we will train custom dense dual-encoder models and build two ANN indexes using the Faiss library. For DIY tasks, which have well-formed titles, we will use a transformer-based cross-encoder re-ranker in a second stage to improve the retrieval quality. We will also develop dense retrieval models that are capable of diversification to provide a diverse set of task instructions to the user. These models will be initially trained on some publicly available datasets (Recipe1M, Quora Paraphrases, etc).

(Multi-Modal) Option Selection: For a given query, we retrieve multiple results. We allow the user to express their choice in natural language, without forcing them to give specific answers. This makes option identification more challenging. To address this, we will first deploy a hybrid matching method that uses semantic embeddings and term overlap to identify the most relevant option. Later, we will train a custom BERT-based NLI model on curated and automatically collected data to estimate a relevance score for every presented option. Our model will be pre-trained first on publicly available NLI datasets (SNLI, MNLI, ANLI). We also plan to ask clarifying questions to the user based on various attributes of the recipes when needed, such as; completion time, calories, dietary restrictions and rating. In addition, we will support image-based option selection through a custom CLIP-based representation model, to allow the user to choose recipes based on their appearance.

(Multi-Modal) Response Generation: We will dedicate significant efforts to response generation. For doing it, we will focus on retrieval-enhanced text generation. We will use FiD-Light [2], a model that is co-implemented by our Faculty Advisor and is currently the state-of-the-art approach in various KILT benchmarks, from QA to dialogue generation.[1] For response generation we will also develop slot filling models based on token classification from the output of large language models. We will also emphasize on multi-modal response generation. For this purpose, we will retrieve images for each instruction and each instruction step, we will generate images (offline) using recent text-to-image models, e.g., Stable Diffusion, and we will retrieve short videos for some instructions that require visual aid. The notion of responses can be different depending on the utterance intent. We will also generate **chit-chat** utterances when is appropriate. We will explore DialoGPT [3], BlenderBot, and web-augmented chit-chat models by training our FiD-Light [2] model. To increase user engagement with the bot, we will also work on system-initiative multi-modal chit-chat conversations, in which an interesting or funny subdialogue is initiated about an image that we present to the users.

(Multi-Modal) Question Answering: Our system will support two types of question answering; in-task QA and open-domain QA. We will use an abstractive transformer-based sequence-to-sequence model to generate answers for a given context. Similar to response generation, our QA model will be based on our state-of-the-art FiD-Light model that is trained on multiple publicly available QA datasets (SQUAD, NQ, NewsQA). Apart from generating the correct answer, we will also explore methods for supporting answers with explanations. Sometimes the user questions can be context-dependent in the scope of a conversation. For this purpose, a sequence-to-sequence query rewriting model will be part of the pipeline to de-contextualize the user utterances. Additionally, we will re-write the QA output to generate well-formed sentences. For example if our QA model returns “2” for the question “how many spoons of vinegar should I use?”, the final well-formed response would be “You need to use 2 spoons”. In the DIY/hobbies domain, some descriptions can be very long and not suitable for a voice-only interface. For this purpose we will use an abstractive summarization model (such as BART) to reduce their length. We will also develop outside-knowledge visual question answering (OK-VQA) models to enable users to interact with the multi-modal content presented to them by asking question about the content of images and videos. Our research group has work on OK-VQA in the past and has experience on this interesting area.

Pipeline 3: Monitoring & Self-learning

Following our vision for customer-focused design, we will implement a human-in-the-loop monitoring and self-learning pipeline. We will use the following signals to automatically identify potential failure points at scale: (1) low rating provided by the user, (2) negative text-based and/or voice-based sentiment of user utterances (3) repetition or paraphrases of the same utterance across user turns (4) system-initiated clarifying questions where the confidence of a model is low. Using these methods, we can identify which system turns are most likely satisfactory and which ones are not. The conversation turns which do not give some negative signal or come after a clarifying question, will be used for improving our existing models over time with limited or no human curation. The potential failure points will be aggregated in a special interface and analyzed over time by one of our team members. If the number of these positive and negative points becomes significant, we might use some stages of external crowdsourcing. Along with the explicit feedback provided by users in the end of each conversation, will will also monitor the negative signals mentioned above. This will give us a more unbiased estimate of user satisfaction even for conversations that ended with no explicit feedback.

Open Sourcing: We will maintain two different repositories. We will develop and maintain a new open-source general-purpose framework for building production-ready conversational agents based on Macaw [4], an open-source tool from our research group. On top of that framework, we will build our TaskBot integrated with the custom APIs provided by Amazon. This will make our work easily re-usable by other researchers and engineers in the future.

References

[1] R. Rombach *et al.*, “High-resolution image synthesis with latent diffusion models,” 2021.

[2] S. Hofstätter, J. Chen, K. Raman, and H. Zamani, “Fid-light: Efficient and effective retrieval-augmented text generation,” *ArXiv*, vol. abs/2209.14290, 2022.

[3] Y. Zhang *et al.*, “Dialogpt: Large-scale generative pre-training for conversational response generation,” in *ACL*, 2020.

[4] H. Zamani and N. Craswell, “Macaw: An extensible conversational information seeking platform,” in *SIGIR*, 2020.

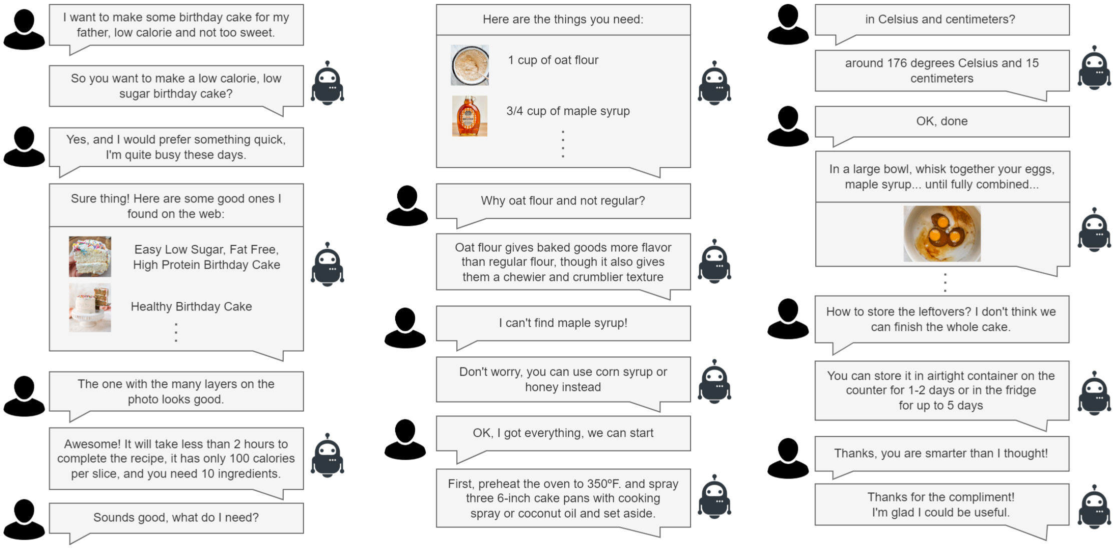
[1] KILT leaderboard: https://eval.ai/web/challenges/challenge-page/689/leaderboard/

Appendix

Lessons Learned from Taskbot Challenge 2021

Our previous participation in TaskBot Challenge was an incredible learning experience. Our proposed team is mainly different from last year (most students graduated or are no longer eligible). Our team leader was part of the last year's team and has substantial experience. This year, with our new team members, we will use the experience from last year, and make even larger research and engineering contributions that this time will be heavily customer-driven. These are the main lessons learned from last year:
From our team: (1) Transferring ideas from research to production takes significant efforts and should not be underestimated. (2) Systematic and methodological monitoring of the system using multiple metrics is very important for debugging, improving our system and increasing customer satisfaction over time. (3) An early strong prototype is very important for the later stages of the challenge. (4) Research projects should be more product-focused and easily transferable to actual features. (5) Careful project planning, strict deadlines, frequent meetings, and pair-programming are vital for the success of our project in a competitive setting.
From other teams: We believe that the great success of other teams was significantly attributed to: (1) effective retrieval using additional data sources other than the provided APIs, (2) generative (open-domain) QA & chit-chat support, (3) multi-modal enrichment by displaying relevant images/videos, and (4) engaging responses with personality. We will take all these lessons into account when making decisions.

An Example Conversation



I want to make some birthday cake for my father, low calorie and not too sweet.

So you want to make a low calorie, low sugar birthday cake?

Yes, and I would prefer something quick, I'm quite busy these days.

Sure thing! Here are some good ones I found on the web:

Easy Low Sugar, Fat Free, High Protein Birthday Cake

Healthy Birthday Cake

⋮

The one with the many layers on the photo looks good.

Awesome! It will take less than 2 hours to complete the recipe, it has only 100 calories per slice, and you need 10 ingredients.

Sounds good, what do I need?

Here are the things you need:

1 cup of oat flour

3/4 cup of maple syrup

⋮

Why oat flour and not regular?

Oat flour gives baked goods more flavor than regular flour, though it also gives them a chewier and crumblier texture

I can't find maple syrup!

Don't worry, you can use corn syrup or honey instead

OK, I got everything, we can start

First, preheat the oven to 350°F. and spray three 6-inch cake pans with cooking spray or coconut oil and set aside.

in Celsius and centimeters?

around 176 degrees Celsius and 15 centimeters

OK, done

In a large bowl, whisk together your eggs, maple syrup... until fully combined...

⋮

How to store the leftovers? I don't think we can finish the whole cake.

You can store it in airtight container on the counter for 1-2 days or in the fridge for up to 5 days

Thanks, you are smarter than I thought!

Thanks for the compliment! I'm glad I could be useful.

Figure 1: Example conversation for completing a recipe, demonstrating retrieval, in-task & open-domain QA, multi-modality, chit-chat and clarifications for robustness & self-learning.

A High-Level Architecture of MarunaBotV2

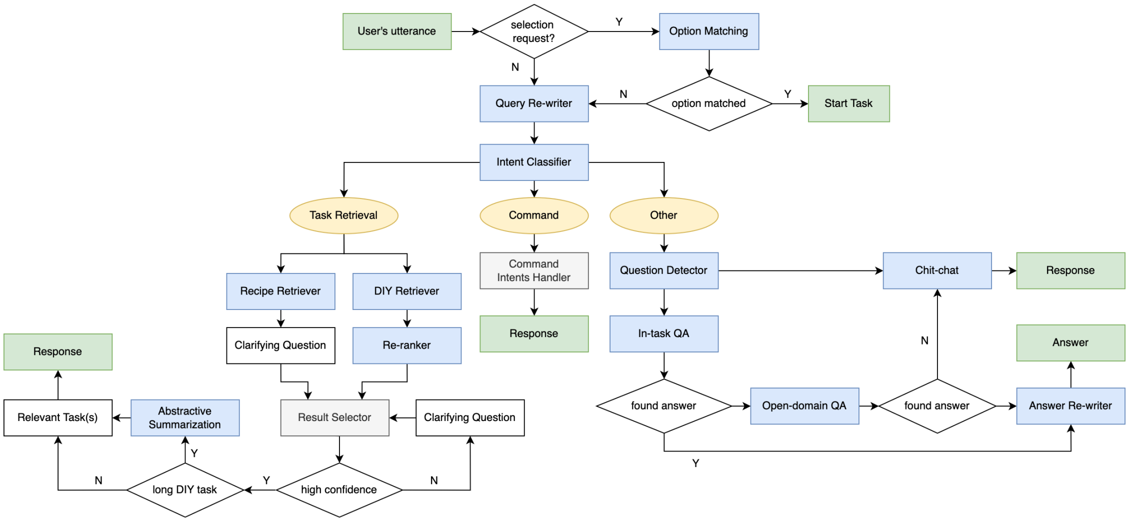


Figure 2: High level architecture of the conversational agent pipeline