THAURUS: An Innovative Multimodal Chatbot Based on the Next Generation of Conversational AI

Marcos Estecha-Garitagoitia[1], **Mario Rodríguez-Cantelar**[2]
[1]ETSI de Telecomunicación, Av/ Complutense 30, Madrid - Spain, 28040
[2]CAR (UPM-CSIC), St/ José Gutiérrez Abascal, 2, Madrid - Spain, 28006
Universidad Politécnica de Madrid
`{marcos.estecha.garitagoitia, mario.rcantelar}@upm.es`

Alfredo Garrachón Ruiz, Claudia Garoé Fernández García, Sergio Esteban Romero, Cristina Conforto, Alberto Saiz Fernández and Luis Fernando Fernández Salvador
ETSI de Telecomunicación, Av/ Complutense 30, Madrid - Spain, 28040
Universidad Politécnica de Madrid
`{a.garrachon, claudiagaroe.fernandez.garcia, sergio.estebanro, cristina.conforto.lopez, alberto.saiz.fernandez, fernando.fsalvador}@alumnos.upm.es`

Luis Fernando D'Haro
ETSI de Telecomunicación, Av/ Complutense 30, Madrid - Spain, 28040
Universidad Politécnica de Madrid
`luisfernando.dharo@upm.es`

Abstract

The next generation of conversational AI has brought incredible capabilities such as high contextuality, naturalness, multimodality, and extended knowledge, but also important challenges such as high user expectations, high latencies, large computational requirements, as well as more subtle problems such as mismatch on existing databases for fine-tuning purposes, difficulties for pre-trained LLMs models to handle dialogue interactions, and the integration of multimodal capabilities.

This paper describes the architecture, methodology, and results of our THAURUS chatbot developed for the Alexa Prize Socialbot Grand Challenge (SGC5). Our proposal relies on several innovative ideas to take advantage of existing LLMs to create engaging user experiences that are capable of handling real users in a scalable way and without compromising the competition rules. Different SotA dialogue generators were fine-tuned and incorporated to give variability and handling the wide range of topic conversations; we also developed mechanisms to control the quality of the responses (e.g., detecting and handling toxic interactions, keeping topic coherence, and increasing engagement by providing up-to-date information in a conversational style).

In addition, our system extends the capabilities of the Cobot architecture by incorporating modules to automatically generate images, provide voice cloning capabilities with fictional characters, serve contextual sounds for detected entities in the dialogue, better capitalization and punctuation capabilities, and to provide natural expressions of interest.

Finally, we also included a trained generative selector and a reference-free model for automatic evaluation of turns that could reduce latencies and complement the ranker's capabilities to select the best generative answer.

5th Proceedings of Alexa Prize SocialBot Grand Challenge (Alexa Prize 2023).

1 Introduction

The goal of interacting in a natural way with a machine has been the goal of many researchers in dialogue systems and natural language processing over the past decades. Starting with ELIZA [1], then moving to more complex assistants such as Alexa, Google Assistant, Cortana or Siri, and then arriving at the more complex models like ChatGPT[1], Claude[2], or BlenderBot vs 3.0 [2, 3]. The future trend is the usage of even larger language models such as Bard / LaMDA [4], GPT-4 [5] or ImageBind [6] that can also handle multimodal inputs, large contextual inputs (e.g., 32k or even 100k tokens), handling tools [7], and even automatically iterate over their partial solutions to obtain better responses (e.g., AutoGPT[3] or BabyAGI[4]).

In spite of recent advances, scalable real-time interactions with real users remain a challenge. On the one hand, as discussed by [8] and [9], the lack of empathy and personalization that users perceive during the interaction, the handling of long-term goals for communication, and the difficulties in handling common sense and reasoning capabilities reduce user engagement. In addition, high expectations created by recent LLMs produce overestimated capabilities in users without considering aspects as hallucinations, generation of long sentences, unexpected errors due to speech recognition errors or because short user's utterances, definition and handling of dialogue states and goals, among others. Finally, the increased number of multimodal Alexa devices (e.g., Amazon Echo Show) has brought interactive screens, which opens new opportunities and challenges that need to be addressed.

Therefore, the purpose we targeted for our Socialbot was to provide an emotional, natural and topic-coherent interaction, capable of handling challenging situations in which users want to express sensitive opinions even in a toxic context, handling multiple generative models with different latencies and capabilities, reducing hallucinations, providing good short and concrete responses in a dialogue style, while also being able to detect when the interaction is not going well by using a self-supervised evaluation metric. Finally, we also included engaging and innovative multimodal modules to produce interactive automatic images using diffusion models [10] and sounds on specific entities; then, we incorporated an automatic evaluation metric on the generated images to detect which ones to display or not. Finally, we also introduced a module for cloning [11] the voice of different fictional characters and merging it with the traditional Alexa voice to allow fun and engaging conversations.

This paper describes our second participation in the Alexa Prize Socialbot Challenge 5 (SGC5) [12]. Starting with our extensive experience in the design and operation of dialogue systems, lessons learned in the process of creating real-user applications [13], and our previous participation in SGC4 [14]. The paper is distributed in the following sections. Section 2 provides an overview of our extended and modified Cobot architecture. Then, in Section 3 we describe the new modules included to extract and process information from user utterances. Section 4 describes in detail the different response generators introduced in the pipeline, including also those that handle user interactions through multimodal interfaces. Section 5 describes our automatic evaluation metric at turn-level. Then, in Section 6 we present the overall results and analysis of the performance of our chatbot. Finally, Section 7 presents our conclusions and ideas for future work.

2 System overview

THAURUS socialbot is built on top of the Amazon Conversational Bot Toolkit (CoBot) [15], which provides a low-effort automated deployment environment for socialbot components with appropriate auto-scaling settings. Cobot uses the cloud services provided by AWS EC2 and ECS, storing user information and chat recordings using DynamoDB. Then, it also handles running external models using Docker containers, storing and serving LLMs through the Elastic File Systems (EFS), or using dedicated cloud-based services (like SageMaker) to create, train, and deploy machine-learning models. Finally, Cobot also allows chatbots to use AWS Lambda functions to allow interacting with multiple users in a scalable way due to that the human utterances create events to which the socialbot reacts.

---

[1] https://openai.com/blog/chatgpt

[2] https://www.anthropic.com/index/introducing-claude

[3] https://github.com/Significant-Gravitas/Auto-GPT

[4] https://github.com/yoheinakajima/babyagi

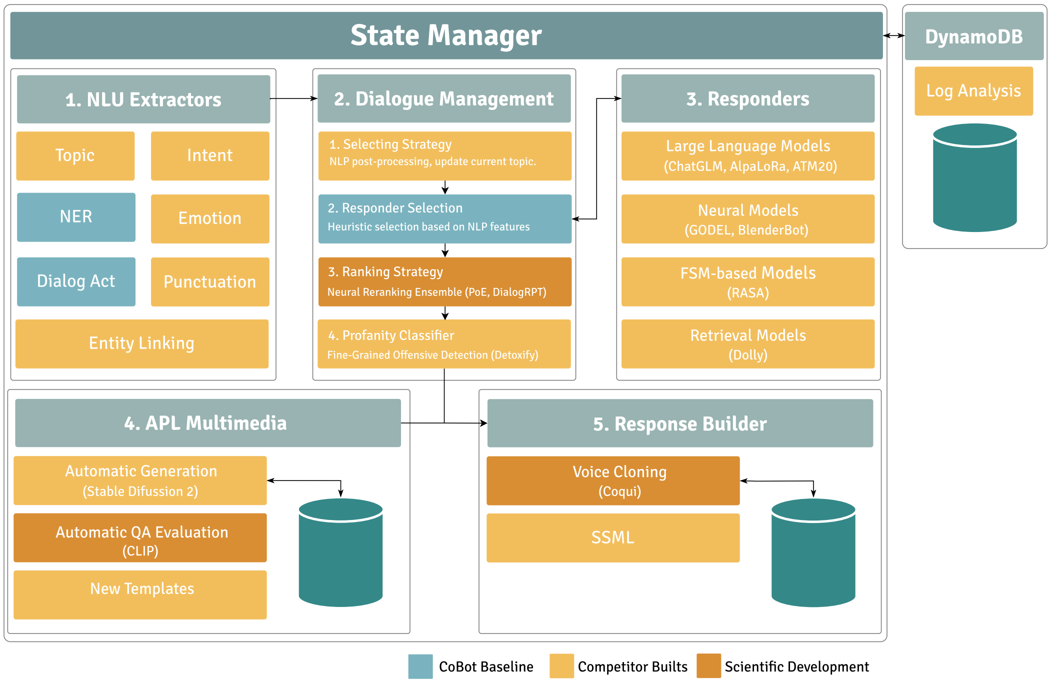


Figure 1: Overall System Design

Figure 1 shows the THAURUS system architecture, indicating the modules we extended or implemented; modules that we will describe in detail in the following sections.

The typical process operation is that when a user connects to our chatbot through the Alexa skill, Amazon's Automatic Speech Recognizer transcribes the user speech and passes us the most likely text transcription, as well as a list of potential hypotheses. Then, our defensive safeguard classifiers and generators (Section 4.6) are used to detect and handle sensitive user requests (e.g. asking for medical or investment advice, opinions about politicians, or trying to provoke a chatbot misbehavior), which are, in most cases, politely handled by asking the user to talk about other topics and providing a list of alternative topics, allowing users to express their opinions without the chatbot providing one, or finally asking users to stop incorrect behaviors.

The ASR transcription is then processed by the Natural Language Processing Pipeline (see Section 3) performing different feature extraction and text classification tasks. The most relevant modules implemented were a topic classifier (Subsection 3.1) and a case recovery and punctuation module (Subsection 3.2). We also made use of other modules, such as the text-based emotion detector using the Vader library [16] and using voice information as provided by Amazon. We also used the built-in Named-Entity Recognizer and complemented it with Spacy[5] and Stanza [17] libraries.

Next, the Dialog Manager selects which response generators will be queried to generate a response. In this case, different generators (see Section 4) were implemented. All of them receive the current turn, previous turns, and contextual information (e.g., knowledge, persona profile, user information, etc.). The response generation manager calls and collects the candidate responses from the different responders. Here, we trained a classifier (see Subsection 3.4) that predicts which generators to call in an effort to reduce latencies and provide better candidates for the response ranker.

Then, the dialogue manager filters out some of the candidates based on detecting unappropriated or sensitive responses, repeated or poor responses, or nontopical coherent responses. In case there is more than one response after filtering, it sends them to the Cobot built-in ranker module, and then the dialogue manager selects the most suitable response based on the information given by the current

---

[5] https://spacy.io/

dialogue state, extracted features, the user's current utterance, current topic, along with the scores provided by the ranker model.

Once the response is selected (in case no response is selected, we implemented several predefined backup responses), the dialogue manager sends it to the Response Builder, which, based on emotions and some rules, adds interjections to the text using predefined SSML tags; lastly, the enriched sentence is sent to the Alexa Text-to-Speech synthesizer. Here, we also added an engaging mechanism to allow Alexa to include portions of a cloned voice in which the Alexa voice imitates the voice of a fictional character (see Section 4.8).

Finally, the interactions and internal states from the different modules in the architecture are stored using DynamoDB. Since the analysis of the logs is a critical component to detect problems and improve the performance of the system, we deployed different scripts and notebooks to analyze the logs, check latency information, but most importantly to combine all interactions, users' ratings, and feedback, to keep our chatbot in control and prioritize improvements.

3 NLU extractors

The natural language understanding modules are responsible for extracting the necessary features from the user's utterance that can be used later on downstream modules. The outputs of the NLU modules are stored in the state manager of the conversation, so other modules can get access to them, but they are also written into the DynamoDB database for posterior analysis. In this section, we will describe the two main new modules we included: a) a zero-shot topic and subtopic classifier, and b) a punctuation and case restoration module.

3.1 Topic and Subtopic Classifier

Since the beginning of our deployment, we considered topic detection as a very important component for the dialogue manager for handling coherent conversations and detecting poor or non-engaging responses from users and chatbot. Although Cobot already provides a built-in topic classifier module that can handle up to 11 different topics (see Table 1), we wanted to have the possibility of increasing the number of labels and providing a more fine-grained detection using subtopics without requiring training or fine-tuning. Therefore, we defined a list of 21 topics and 116 subtopics (6 on average per topic) and used the approach proposed in [18] in which pre-trained NLI models can be used as ready-made zero-shot sequence classifiers. Here, the classification is done by posing the sequence to be classified as the NLI premise and by constructing a hypothesis from each candidate label (in our case, an element from the list of topics or subtopics).

Table 1: List of labels predicted by the CoBot built-in topic classifier.

1. general	2. books	3. interactive	4. inappropriate	5. sports	6. science
7. politics	8. phatic	9. music	10. movies	11. *other*	

To this end, we proposed a novel hierarchical approach to detect and classify topics and subtopics in a conversation [19]. This module provides detailed topic and subtopic information for handling the dialogue management, for instance, response generator selection, ranking strategy, and dialogue strategy to steer the conversation in one direction or another (topic switching). This work was expanded in [20]. This system uses a NLI-based zero-shot model that could be used hierarchically to detect the high-level topic and then the subtopic. To do this, the BART-Large model [21] was trained on the MultiNLI (MNLI) dataset. To test the capabilities and scalability of this classifier, we tested it on a subset of our collected interactions during SGC5.

Results The SGC5 subset was first classified with the BART model using the same 11 classes used by the CoBot topic classifier (Table 1). Both classifiers agreed in their results with an F1-Score of 0.26. This low score was partially due to the large number of times that the CoBot topic classifier labeled an utterance with the class *other* (i.e., 63.92% of the turns), while the BART model only used it 24. 91% times. With this in mind, we classified the entire SGC5 subset with the 21 topics proposed in Table 2, and analyzed them to see which model provided the best result. In Figure 2 we analyze how the topics were distributed in the SGC5 subset.

Table 2: List of topics and subtopics to detect using our Zero-Shot NLI-based Approach.

No	Topics	Subtopics
1	animals	cats, dogs, pets.
2	art	ballet, cinema, museum, painting, theater.
3	astronomy	galaxy, planets, stars, universe.
4	books	author, genre, harry potter, plot, title.
5	education	history, mark, professor, school, subject, university.
6	family	children, friends, marriage, parents, relatives.
7	fashion	catwalk, clothes, design, dress, footwear, jewel, model.
8	finance	benefits, bitcoins, buy, finances, investment, sell, stock market, taxes.
9	food	drinks, fish, healthy, meal, meat, vegetables, dessert.
10	movies	actor, director, genre, plot, synopsis, title.
11	music	band, dance, genre, lyrics, rythm, singer, song.
12	news	exclusive, fake news, interview, press, trending.
13	photography	camera, lens, light, optics, zoom.
14	politics	biden, fake news, elections, poll, trump, vote.
15	science	math, nature, physics, robots, space.
16	social media	facebook, fake news, instagram, twitter.
17	sports	baseball, basketball, coach, exercise, football, player, soccer, tennis.
18	vehicle	bike, boat, car, failure, fuel, parts, plane, public transport, speed.
19	videogames	arcade, computer, console, nintendo, play station, xbox.
20	weather	cloudy, cold, hot, raining, sunny.
21	*other*	

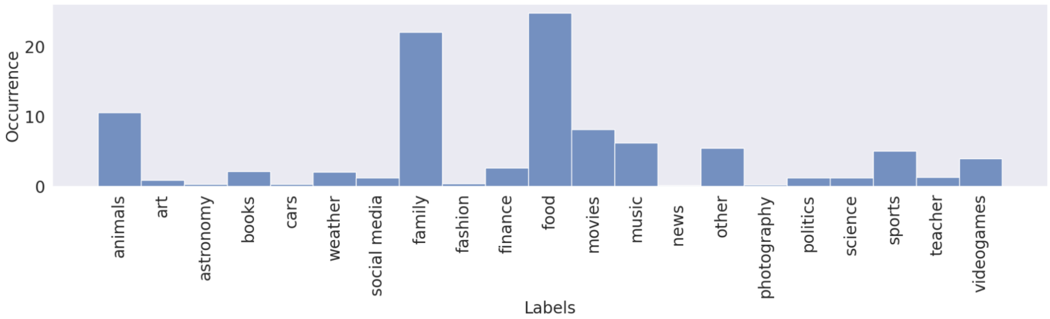


Figure 2: Occurrence of the 21 topics in the SGC5 subset classified with the BART model.

As can be seen in Figure 2 the most commonly occurring topics are: animals, family, food, movies, music, sports, videogames and *other*, representing 86.14% of the automatically annotated turns.

In this case, only 5.49% of the cases were labeled *other* by the BART model when using 21 topics, showing an important advantage over the built-in CoBot topic classifier that only uses 11 topics. Thanks to this, we have a higher granularity of the turns of the dialogues, which brings us a more dedicated and focused conversation on topics.

Once we knew in more detail the topic the user wanted to talk about, we went down to a more detailed level of granularity within the topic by also classifying at subtopic level. Table 3 shows the 10 most recurrent subtopics labeled by the BART model in the SGC5 subset. Of the 116 subtopics, only 89 subtopics were detected by the BART model in the selected SGC5 dataset. 73.22% of the turns shared the subtopics friends, healthy, pets, gender, meat, console, relatives, gamer, parents and song.

Latency The required times needed by the BART model to classify a turn into different topics and subtopics are presented in Table 4. The table shows the average time it takes for the model to classify 10 distinct sentences. It is evident that the processing time increases as the number of classes to analyze increases. However, the processing times for all scenarios are relatively short, with an average of 39 ms required to select from the set of 21 different topics shown in Table 2. This demonstrates a high scalability of this system, allowing a larger number of topics and subtopics to be used to

Table 3: Occurrence of the top 10 subtopics in the SGC5 subset classified with the BART model into 116 different topics.

Subtopic	**Occurrence (%)**	**Topic it belongs to**
friends	16.39	family
healthy	16.22	food
pets	10.96	animals
genre	8.09	books/movies/music
meat	7.08	food
console	3.20	videogames
relatives	3.10	family
player	3.06	sports
parents	2.80	family
song	2.32	music

classify the turns of conversations with little increase in the response time of the overall system. Measurements were obtained using a computer equipped with an Intel(R) Core(TM) i9-10900F CPU @ 2.80GHz. The code execution was performed on a NVIDIA GeForce RTX 3090 24GB GPU using CUDA version 11.4.

Table 4: BART classifier latencies for classifying topics and subtopics in the SGC5 datasets.

Dataset	**Granularity**	**#labels**	**Average Time (ms)**
SGC5	Topic	11	27
SGC5	Topic	21	39
SGC5	Subtopic	6	19

3.2 Punctuation and Capitalization Restoration Module

Another module we implemented was an alternative to the built-in punctuator provided with Cobot, which also has limited capabilities for casing restoration. Our motivation was to guarantee a higher correct detection of punctuation marks such as question and exclamation marks, together with a low latency and being able to capitalize nouns in order to improve the performance of other modules such as sentiment analysis, named-entity recognition, entity-linking, among others.

After testing multiple options, we opted for using NeMo [22], a toolkit developed by NVIDIA to create conversational AI models, which offers a really efficient and pre-trained Punctuation and Capitalization Model. By default, the model supports the restoration of commas, periods, and question marks. It can also predict which words should be capitalized or not. The NeMo model is based on BERT [23] and is jointly trained on two token-level classification tasks (sequence labeling and capitalization detection).

Before implementing the model into our pipeline, we also measured its performance against the built-in model included in Cobot on 100 randomly selected sentences that were manually annotated. Table 5 shows the comparative results. In the table, the Word Error Rate (WER) was obtained using the Evaluate library[6]. We include two results: a) one for comparing just the punctuation task, and b) including also the restoration of the capital letters (the capitalization task). As we can see, both models are good, but the F1-score for NeMo is slightly better, and when considering the capitalization task the performance is higher, which is important for other tasks in the pipeline.

3.3 Detection of User-Bot Direct Interaction and Toxicity using Quantum NLP

One of the problems we detected during SGC4 and SGC5 when handling toxic users is that the responses of most chatbots were to forbid the user to continue talking about the detected sensitive topic. This was done to avoid engaging the chatbot in potentially compromising conversations.

---

[6] https://huggingface.co/docs/evaluate/index

Table 5: Performance comparisons between NeMo vs. the built-in punctuation model. The results in brackets for the F1-Score correspond to point, comma and question mark respectively. WER results shows when only punctuation restoration or when also capitalization were considered.

Model	Precision	Recall	F1-Score	WER
NeMo	95.92%	89.24%	92.46% (96.6%, 82.7%, 100%)	2.24% / 4.9%
Punctuation API	92.56%	86.71%	89.54% (96.9%, 71.7%, 100%)	4.17% / 20.50%

However, when we manually checked several of these cases, we detected that many times the user: a) wanted to express their opinion about the topic, not to ask a specific response from the chatbot, or b) the user was using sensitive words but that were not addressed towards the chatbot, but to refer to external people or situations. In order to reduce these false positive situations, we decided to implement a module that could detect: a) when the user is addressing our bot directly or not, and b) to detect the kind of interaction that the user is having, if it is a neutral comment, a toxic attitude, or a sensitive topic. Then, our purpose was to create a multiclass classifier whose output could be handled through the Eliza pipeline (see subsection 4.6).

Although this kind of multiclass classifier can be easily implemented using pre-trained Transformer-based models, we also wanted to test the possibility of exploring an innovative path by using Quantum Natural Language Processing (QNLP) techniques. QNLP is a relatively new field that can generally be described as a discipline that looks to leverage the properties of quantum computing (superposition, entanglement, and quantum interference) to process and analyze human language data. Since it is quite novel, not many tools have been developed. One of the few, but most actively developed, is Lambeq [24]. This is an open-source high-level Python library created by the Quantinuum QNLP team. One of the advantages of the library is that it can be run in quantum computers, but also offers a neat package for performing quantum simulations in traditional hardware. Therefore, we tested its capabilities in hope that in the future it can be run in Quantum computers with very low latencies and take advantage of all its potential computing power processing.

In order to create such a model, we first created a custom dataset that fits exactly the types of category that we wanted to cover.

1. Label '0': The user is expressing a neutral opinion but NOT directly targeting the chatbot.
2. Label '1': The user is expressing a neutral opinion but directly targeting the chatbot.
3. Label '2': The user is expressing a Toxic/Sensitive opinion but NOT directly targeting the chatbot.
4. Label '3': The user is expressing a Toxic/Sensitive opinion and directly targeting the chatbot.

The sentences for this dataset were randomly selected from conversations of our chatbot with different users considering those that were detected as sensitive, labeled with the intents: InformationRequest, Opinion_Expression, Opinion_Request and General_Chat by the built-in Cobot Intent Classifier. To reduce biasing the classifier, we equally distributed the annotated sentences between the 4 categories.

Then, following the pipeline described by the Lambeq team, we trained a binary model using the first two classes, which reached an accuracy level of 79% and, secondly, a more complex multiclass model covering all four categories, which reached an accuracy level of 49.5% (which is almost twice the random choice). Considering the limited size of the dataset, these results seem to be good enough. Our next goal is to optimize and fine-tune the multi-class model further and also to develop a full module around the quantum model.

3.4 Responder Predictor Module

One of the core elements in our chatbot architecture is the usage of generative models to handle the dialogue in a scalable way. However, calling all generative models will introduce latencies due to the process of collecting and then ranking the retrieved responses. To reduce this, we trained a classifier that could provide an a priori prediction of which responders must be called given the

Table 6: Classification performance of the trained model to select the best generator.

Dataset	Accuracy	f1-weighted	f1-macro
Prediction of the selected chatbot	0.54	0.48	0.36
Prediction of the annotated chatbot	0.60	0.56	0.38

context of the dialogue. The selected responders will be used together with a set of backup responders to avoid passing no options to the ranker. To train the classifier, we relied on dialogues with high user ratings (rating higher than 3.5) and large number of turn interactions (higher than 10 turns) collected during the beginning of the competition. In this way, the classifier will capture the rules and dialogue strategies that produced the best scores.

Modeling To keep the process fast (low latencies), for training, we took into account the previous turn and the current utterance provided by the user. These turns were labeled with "[USER]" and "[BOT]" tags. Since user ratings are at the dialogue level, when selecting a dialogue for training, we assume that all selected responders involved in those conversations contributed equally to obtain such a score. Furthermore, since some responders were selected more than others, a data balancing process was also implemented to avoid the classifier learning a biased distribution of the data. In addition, to evaluate the model, some of them were manually annotated as the labels used for training come from the ranker, which may not select the best response among those available. In this way, it is possible to also evaluate if this model would help to provide better responses.

On the other hand, since the ranker is a neural model (a fine-tuned BERT-based model over 8k turns from previous Alexa Competitions) and human language is actually complex, so it is possible that the selected responders were not the best among those available. In that case, it is assumed that we are training with noisy data. Consequently, a manual annotation was carried out in order to check the performance of the model over controlled data.

Once the data was prepared, a simple fine-tuning process was performed to test the effectiveness of the developed solution. In particular, we fine-tuned the ”distilbert-base-uncased”[25]. As 17 labels have been considered, the probability of obtaining a random prediction is $1/17$ is around 5.88%.

Results Table 6 shows the performance of the model when comparing the selected generator in the actual turn vs. the one proposed by the classifier. Then, we also show the results when using as ground truth the labels from the manually annotated dataset with 255 samples. It can be seen how results are above the probability of randomly selecting one responder so it can be assumed that the solution is working as expected. Something that can be noticed is that the results obtained on the annotated dataset are slightly better than the actual selected one. As future work, we want to train a multi-label classifier to allow selecting more than one responder. This idea will require more manual annotations and establishing a threshold to decide which responders pass and which ones do not.

4 Responders

One of the most important components in our proposed architecture is the incorporation of high-quality responders that can provide answers to the large variability of questions, opinions, and requests from the user. Here, the trend during previous editions of the Alexa Prize Socialbot competition was to use hybrid approaches by combining rule-based, state-based, or neural generative approaches. Following our approach during SGC4, and considering the large improvements when using LLMs, we also relied mostly on neural generators (see section 4.1), some hybrid approach when the user is asking for some specific information, in which the chatbot needs to get access to some specific knowledge or retrieve it (for instance, to provide recent information about movies or news, see sections 4.3 and 4.4). Then, we also implemented some state-based models (see sections 4.5 and 4.6) for getting personal information from users (e.g., name, topics that like or not to talk about, etc.). Finally, we also implemented some hand-made responses (see Subsection 4.7) for handling switching topics, providing fun facts, ephemeris, or jokes.

4.1 Neural-based Response Generators

The development of neural models for generating responses in open-domain conversations is a continuous and thrilling endeavor within the field of natural language generation. Recent improvements in terms of high and larger amount of training data, instruction fine-tuning, incorporation of human feedback, fast mechanisms for fine-tuning (e.g., PEFT), and quantization approaches, among others, have brought an unprecedented usage of large language models as response generators in chatbots. Unfortunately, there are still many issues to solve, including problems to keep up-to-date the information learned by the model, the occurrence of hallucinations, toxic or biased responses, mismatches between the style of answers provided by a LLM vs the dialogue style required in the competition, as well as high latencies that make difficult to solely rely on LLMs.

In our approach, we tested more than 25 different pre-trained SotA chatbots. For each, we first performed some manual tests to check for speed, quality of the responses, and difficulties for integrating or fine-tuning such a model. Then, we included them in the A/B testing and Experimental settings provided by Cobot, and checked the average ratings provided by the users to all dialogues in which the response generated by that chatbot was presented to the user. Finally, we also classified chatbots by topic, allowing us to decide which chatbot to use depending on the detected topic.

Finally, we only used a total of 10 generative chatbots. Several of them provided by Amazon, such as Neural Response Generator (NRG), Topic_NRG, or Empathy_NRG, which are GPT2-based model fine-tuned on different datasets including some previous years interactions with Alexa users). Then, we also included LLMs such as AlexaTM 5B and AlexaTM20B (introduced in [26] and pre-trained in more than 1.5TB of data and fine-tuned in 9 conversational dialogue data). We finally also included ChatGLM-7B [27], BlenderBot 2.7B [28], GODEL[29], Dolly [7].

Among those models that we tested but finally discarded (mainly due to high latency or difficulties in fine-tuning them), we can mention the following: BlenderBot 3B [2], Vicuna [30], and Alpacoom [31].

4.2 Generative Model for Handling Movies

The objective of this generative responder is to answer user questions about movies in a precise way with up-to-date information. For this purpose, we used GODEL [29], a model specifically pretrained on conversational data; but unlike other models, its main goal was to support downstream dialogue tasks that require external information to the current conversation and an easy adaptation process. In our case, we included it to have a generative model capable of responding in a conversational style to questions about movies (even very recent ones). For this, we fine-tuned GODEL with the data of conversations automatically generated by ChatGPT [5], containing information about the movie in the prompt, and then retrieving the movie knowledge from two movie APIs.

Implementation We first generated appropriate conversations on the specific topic (movies in this case, but extensible to other domains) for fune-tuning purposes. So, we first retrieve movie data using two different APIs (IMDB and TMDB). The data obtained was then incorporated into a predefined prompt template that was used to generate a dataset of synthetic dialogues generated with ChatGPT (see some examples in Annex 1 7). Due to the fact that the GODEL model generalizes well due to its few-shot capability, it was not necessary to obtain many conversations (in our case, we just collected +160 excepts) to obtain good results during the fine-tuning task.

Once the model was ready, it was integrated into the Cobot architecture and logic, by considering that it should only be called when the subject of the conversation is about the specified topic and, in turn, being able to identify the entity about which information is required from the user's request (in this case, the movie in the utterance). Finally, to reduce latencies, the system stores the movie information so that when the same or another user is talking about the same movie, it is not necessary to make a new request to the APIs. Figure 3 shows the pipeline for training and runtime processes.

Qualitative results The model was fine-tuned after 50 iterations with 130 excerpts from the conversations generated using ChatGPT (with their corresponding knowledge) and 30 excerpts for both validation and testing. With the model already prepared, a series of subjective tests were carried out on the test data. Our results showed that the model was able to adapt very well to the different fields provided as knowledge and, therefore, to the dialogue style and to respond concretely and

---

[7] https://github.com/databrickslabs/dolly

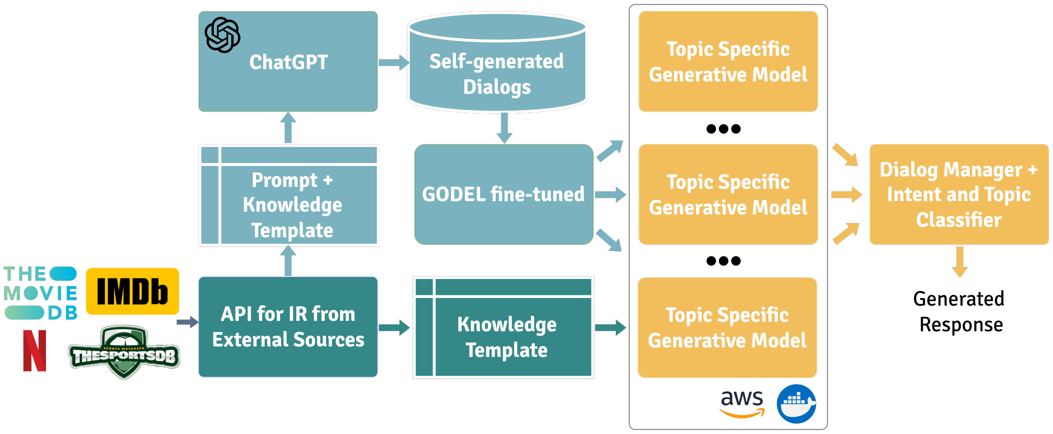

ChatGPT
Self-generated Dialogs
Prompt + Knowledge Template
GODEL fine-tuned
THE MOVIE DB
IMDb
THESPORTSDB
API for IR from External Sources
Knowledge Template
Topic Specific Generative Model
Topic Specific Generative Model
Topic Specific Generative Model
aws
Dialog Manager + Intent and Topic Classifier
Generated Response

Figure 3: Pipeline for the proposed solution. In green the training process, in blue the generic aspects, and in orange the process at runtime.

correctly to the questions about the movies asked by the user (see Annex 2 7). To exemplify, a comparison has been made between the answers obtained using the original GODEL model and the refined model (see Annex 3 7). As we can see, the original model was good but was unable to segment the knowledge data well, mixing the data and giving answers that did not make sense. On the contrary, the refined model learned to make use of the knowledge providing answers to the user and keeping the conversation engaging.

4.3 Generator for handling QA Knowledge

This responder was introduced to answer all user's questions related to knowledge in the most informed and accurate way possible, but keeping a conversational style. For this purpose, two different models were used: the DOLLY model [32] in charge of providing the requested knowledge, and GODEL [29] model to transform the knowledge information into a conversational style.

Implementation In order to answer knowledge questions, the first step is to detect them. Thanks to the high accuracy of our punctuation model (see section 3.2) we mostly relied on turns including the question mark "?" at the end of it, and we combined it with the built-in QA detectors provided by Amazon in Cobot (which include a QA factoid binary classification model and a QA relevance classification model which classifies a response as relevant or irrelevant to a question). Additionally, we slightly increased the priority of this generator in the ranker to avoid the response of a more generic responder to be selected.

Before selecting DOLLY, we tested multiple models (including Macaw [33] and Atlas [34]). However, we found that Dolly-v2-7b model provided excellent responses from the user's own prompt and without the use of contexts or entity recognition. This model, as a result of its training, identifies what is being asked and provides the necessary information in a concise manner as an answer to the question. Then, GODEL takes the provided knowledge from Dolly together with the 3 previous turns and then generates a response in a user-friendly conversational format.

Results After manually testing multiple questions received by our chatbot during the competition and confirming that most of the answers provided by DOLLY & GODEL were good. We focused on reducing the latency of calling both. Here, we established an upper latency limit of 2.5 seconds. In our measurements, GODEL took on average 0.7 seconds to generate a response, leaving us with a free margin of 1.8 seconds for the DOLLY model. This model has an internal adjustable parameter to limit latency (and, therefore, the max length of the generated knowledge response), which we set to 1.5 seconds to leave a free margin of 0.3 seconds in case there could be any additional delay in calling the modules on the cloud. In most cases, DOLLY delivered the answer in less than the indicated time, but it took longer in cases such as being asked about a person's biography, as the model went into a lot of details and generated longer responses.

Table 7: A summary of the main statistics related to the launch module considering conversations where the module appears against those where it does not.

Module	**Mean rating**	**Std. deviation**
Launch module	3.11	1.52
W/o launch module	2.94	1.55

4.4 Generator for Handling News

This responder was introduced with the goal of handling situations in which users wanted to obtain information about current events or incidents. This responder follows a similar approach to the knowledge responder (see section 4.3), using as information source the News Retrieval API provided in the Cobot architecture and the GODEL model [29] to generate the final response.

Implementation To detect the intent in which the user asks for news, we manually created several RegEx. Then, we collected the entities detected in the user's utterance from the NER (named entity recognition) model. From these entities, the most relevant ones are selected, and with them a request is made to the News Retrieval API, which returns a series of news with both their headline and summary. Then, we performed some prompt engineering to find the most appropriate instruction to send to the GODEL model so that it could segment the different news items and not mix their information. Here, similarly to the Knowledge responder, we send the news information as the "knowledge" parameter and the context of the conversation composed of the last 3 turns of the user and the chatbot. Taking into account the possibility that the user remains interested in the news, we keep both the news information and the reply in the reply pool for the next turns.

Results We manually tested several responses generated by this module and found that it was able to segment the news information well and provided a good quality response related to the user's request. We also measured the average latency and found it be less than 1.5 s.

4.5 Launch Responder based on RASA

One of the key points of a conversation is its beginning. We realized that assigning this part to neural responders may not be optimal, as they produced salutations including invented names or preferences for the users creating a bad user's experience and also affecting the rests of the turns. To solve this, we developed a module based on a finite state machine (FSM) which was in charge of the first conversation turns and for extracting some user's personal information (name, like and unlike topics, etc.), that could ensure a proper beginning and contribute to make a more enriching conversation for the user. The definition of states and transitions was done using the RASA framework. In the definition of the states, we also considered the cases where the user does not want to provide such personal information, as well as the cases for proposing some topics to break the ice.

Implementation To ensure that this module starts the conversation, we prioritize it over the other generative models. However, in case of errors, the normal flow of asking other generators and ranking them will be done. Besides, to allow sharing information with the rest of the generators, all turns handled by RASA are also stored in the history of the conversation and the extracted user's preferences are included in the user's table.

In order to train the intent classifier used by RASA to transit between states, it is required to provide examples of when each specific intent is triggered. To speed up the creation of such examples, we used ChatGPT, starting with some few manual examples and prompts, then collecting a total amount of 275 sentences balanced among all the states considered. Then we carefully reviewed them to ensure that they fit with the corresponding intent.

We also included a mechanism to handle cases where users do not provide their real names or the ASR does not correctly transcribe them. So, we created templates and combined them with regular expressions (RegEx) to identify the words corresponding to the provided user's name. Then, we checked the extracted name with a large dictionary of names to ensure that it is an actual name. If the name is not detected as such, the module will try to request it again. If after an attempt, the name was not correctly captured, the chatbot will continue the conversation.

Results To evaluate the performance of this module, we used an A/B test, comparing those user ratings in which the module is called vs those in which it is not. Although these ratings include the influence of other responders along the dialogue, in principle we could see the trend of its impact. Table 7 shows the comparative results showing a slight advantage when using our Launch module. The results were obtained after analyzing data from March to May, inclusive.

4.6 Generator for Handling Toxic Users

In the context of open-domain conversational systems, handling sensitive topics is an important aspect that can be approached in different ways. Some examples of such situations are when providing opinions, especially about controversial topics, or when using swear words but without the intention of misbehaving. One potential strategy involves the identification of such situations and then redirecting the conversation to alternative topics to avoid potential issues. However, in real-life conversations, when someone expresses opinions (even if they are controversial or not), we theorize it could be beneficial to engage with the viewpoint to ensure the user feels heard, without necessarily supporting the expressed ideas or providing any feedback. Therefore, we consider it important to accurately detect when opinions or toxic comments are directly targeting Alexa vs. those targeting third parties.

For the implementation of these ideas, we decided to take inspiration from the Eliza chatbot [1] developed by Joseph Weizenbaum, implementing a Rogerian psychoterapist model. This model facilitates the open expression of users' opinions while avoiding providing personal opinions. In our case, we take the same approach to handle controversial topics, allowing users to express their opinions and concerns about others, so they can unburden themselves.

For implementation, we designed an FSM solution using RASA (similar to the Launch module in Section 4.5) in combination with Detoxify to identify when to use this module. For the RASA module, we also provided several examples to train the intent classification model responsible for the transitions inside the FSM. In this case, ChatGPT was used in the same way, obtaining a total of 1083 samples that were approximately equally distributed among all possible states. Then, a manual review was performed on all generated samples.

Then, we were required to include two classifiers: one for detecting toxic comments and another one for detecting the intentionality and target of the user comments and opinions. In the following, we describe the solutions implemented.

Detoxify The detoxify[8] model uses transformer models for toxicity classification. The model is capable of identifying general and identity toxicity. Table 8 shows up to 16 labels that it is capable of detecting. Thanks to this fine-grained labeling classification, we are able to handle a wide range of scenarios.

Table 8: All the categories considered in Detoxify module.

List of Detoxify labels				
General	1. toxicity 5. insult	2. severe toxicity 6. threat	3. obscene 7. sexual explicit	4. identity attack
Identity	8. male 12. jewish 16. psychiatric or mental illness	9. female 13. muslim	10. homosexual, gay or lesbian 14. black	11. christian 15. white

To illustrate the need to properly recognize these types of toxicity, Figure 4 is presented after collecting data from March to May, inclusive. Here, the results show that a large proportion of turns (almost 20%) were identified with the toxic label, which is a large number and therefore important aspect to handle.

SetFit For the classifier in charge of identifying the target of a comment, we fine-tuned a pre-trained SetFit classifier [35]. SetFit model is an efficient and prompt-free framework for few-shot Sentence Transformers. The pre-trained Sentence Transformer is fine-tuned on a small number of text pairs, in a contrastive Siamese manner. The resulting model is used to extract rich text embeddings, which are then used to train a classification head. This approach achieves high accuracy with orders of magnitude fewer parameters than the existing techniques, therefore providing a robust model for real-

---

[8] https://github.com/unitaryai/detoxify

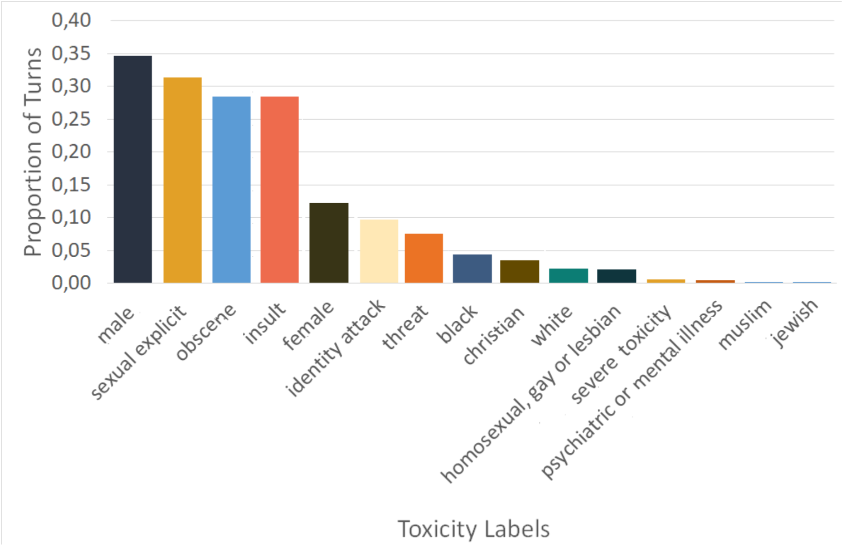


Figure 4: The proportion of toxic comments considering those turns with ratings available

Table 9: A summary of the main statistics related to the Rogerian chatbot module considering conversations where it appears

Module	Mean rating	Std. deviation
With Eliza module	3.328	1.415
W/o Eliza module	2.949	1.545

time applications. In our case, we extended the custom dataset that was used for the quantum module 3.3, reaching more than 1k manually selected sentences from our bot, and adding two additional labels. The six resulting categories were the following:

1. Neutral Comment/Opinion of the user NOT addressing the chatbot directly.
2. Neutral Question/Request directly towards the chatbot.
3. Controversial or Sensitive Comment/Opinion of the user but NOT addressing it to the chatbot.
4. Sexual Comment/Opinion of the user NOT addressing the chatbot directly.
5. Controversial or Sensitive Question/Request directly towards the chatbot.
6. Sexual Question/Request directly addressed towards the chatbot.

Qualitative Results For fine-tuning the SetFit model, we split our data into 70/30, that is, over 700 sentences were used for training and 300 for testing. The trained model achieved an accuracy level of 93.2% for the six categories, with an individual score of 93.8% for label 1, 94.6% for label 2, 92.2% for label 3, 93.7% for label 4, 95.2% for label 5 and 89.1% for label 6. The model was then included in a docker to incorporate its predictions into the Eliza pipeline.

Overall Qualitative Results When we consider now the particular distribution of the ratings obtained from the conversations in which this module was used, the results show a clear advantage showing how the effect of allowing users to express themselves and also reducing sensitive false positive detections contribute to better ratings (see Table 9). The results presented belong to data collected from April to May, inclusive.

4.7 Handling Topic Transitions and Uninterested Users

In order to ensure engaging conversations, it is necessary to detect scenarios in which users are not engaged with the chatbot. For example, a decrease in user interest or a negative attitude towards chatting can be good indicators that the conversation is about to end. This is why two different modules were developed with the aim of re-engaging the user. They focus on detecting situations where a change of topic is needed or providing alternatives that may be more pleasing to the user.

Here, we implemented two potential strategies for detecting user's disinterest:

1. Detect expressions or keywords in the user's utterance that are related to the loss of interest or a negative connotation through a ReGex scheme. Some examples are sentences like "I don't care" or "whatever".
2. Consult the state of the conversation in the last 5 turns; for this, we make use of the sentiment analysis module executed in each interaction with the bot. If we detect that in 3 of the last 5 turns the sentiment is negative, we considered it that the user is not interested in the course of the current conversation or does not like the topic being discussed.

This logic complements the one in which the user directly asks the chatbot his or her desire to change the current topic or request a specific one.

Topic selection There are three possibilities when it comes to change the topic. First, the chatbot suggests a change to some of the topics that have not been discussed with the user before. If all topics have been tried in previous turns, then the chatbot will suggest the ones that were discussed first in the history and then reset the list. The second option is to let the user propose the topic, so we get a more personalized conversation for the user. Then, the third option is what we call a "surprise" topic, which is explained next.

Jokes, ephemeris, fun facts and voice cloning The aim of this module is to liven up the conversation and re-engage the user's interest in the conversation. Here, we implemented an FSM that involves multiple turns to ask the user which of the "suprise" options (which include providing jokes, ephemeris, some fun facts, or allowing voice cloning) is interested in. There are two possible options after asking the user to select one of the "suprise" options:

1. If the answer is a positive response or it contains a positive statement, the corresponding "surprise" topic is provided on the next turn and some feedback question is provided at the end to receive user feedback about the selected topic.
2. If the answer is a negative response or it contains a negative statement, the corresponding "surprise" topic is not provided and the user is asked to indicate a topic of interest.

4.8 Voice Cloning

The need to create a more immersive and personalized user experience requires the creation of emotional connection between users and their digital assistants. Therefore, we have developed an innovative voice cloning system for fictional characters that aims to enhance the user experience by seamlessly integrating with the existing Cobot infrastructure. The goal is to clone voices that are recognizable to the general public in a way that improves the user experience by combining Alexa voice with the voice of selected fictional characters (we preferred to use fictional characters to avoid legal issues with actual people). In this way, by integrating it into our offtopic system, we can improve the user's Quality of Experience (QoE) through the use of surprise and familiarity with the characters.

For implementation, we used the Coqui TTS workframe[9] as the central pillar of our cloned speech generation pipeline. Therefore, a text-to-audio multi-speaker model was trained using the Coqui tools to obtain the cloned voices. Coqui TTS is built on a deep learning architecture that employs a combination of generative adversarial networks (GANs) and variational autoencoders (VAEs) for high-quality voice synthesis. Several last generation models were tested, such as VITS [11], YourTTS [36], or Tortoise [37] models; selecting the VITS model as the best.

The module requires a robust dataset consisting of voice samples from fictional characters, which are obtained through various media sources, including movies, TV shows, or YouTube videos. This dataset must be processed using several audio filters and audio source separation models to separate the voices of the selected characters from the rest of the sounds in the original audio. Besides, in order to make the Alexa Voice to be mixed with the voices of the fictional characters, we also needed

---

[9] https://github.com/coqui-ai/TTS

to clone Alexa Voice and then create the mixed voice by combining both clone voices in different proportions.

Model For training the models, we collected 3.5k audios per speaker. Episodes in the data set were subjected to the following automatic processing to increase the quality of the training data.

1. Demucs Music Source separator [38] was applied to remove noise and music from the selected videos.
2. Each audio was then diarize using PyAnnote [39] to reduce the effort on manually selecting the audio samples for the same speaker.
3. Then, the Whisper ASR [40] was used to align the voice with the transcribed text and to discard segments that are not clear (low confidence).
4. FFMPEG-Normalize (with 22050 Hz Mono) was used to normalize the collected audios in a single sample rate.

Once the character dataset was processed, we trained the voices using Coqui TTS. In our case, we used 8k epochs per character on a single GPU. In addition, during the training process, we specified a mixing percentage between Alexa voice and the character's voice so that the cloned voice does not contrast excessively with the rest of the conversations (spoken using Alexa voice).

Response Generation and Selection The end-to-end generation pipeline is an essential aspect of the system's functionality, ensuring seamless integration with the existing AWS infrastructure. The pipeline consists of the following components:

1. Our chatbot sends an API call with the data containing all the necessary information (user utterance and selected clone voice, in case the user wants the clone voice to say something specific) that needs to be processed by the TTS model.
2. The trained Coqui TTS model synthesizes the combined fictional character's voice with Alexa's voice and returns the WAV file audio via the API call response.
3. Due to the peculiarities of the SSML <audio> tag [10] (48 kbps bitrate and 22.05 kHz, 24 kHz or 16 kHz sample rate in the .mp3 container), it is necessary to perform a conversion of the .wav file using ffmpeg.
4. The mp3 file is uploaded to an S3 bucket. That S3 bucket has a CloudFront distribution associated, so the mp3 is accessible from our chatbot and played to the user.

From our experience during SGC4 while making modifications to Alexa Voice using SSML tags, we know that it could create confusion. Therefore, to make the new voice fit naturally into the conversation and not provoke rejection or confusion in the user, it was decided to indicate to the user that Alexa "knows how to imitate" the character in question, and to show it to the user in a sympathetic way. Additionally, the cloned voice was also surrounded by an initial phrase and a final phrase with the typical Alexa voice. Introductory and concluding sentences are done by selecting one of several predefined sentences to provide flexibility to the conversation. When possible, voice generation was also accompanied by an image of the fictional character using the APL language to further enrich the user experience.

Integration of multispeaker model in bot To improve user interactivity and achieve greater immersion in the cloned voice experience, a three-layer system was designed to integrate our innovation into the conversation. This system is triggered either by regular rules that require the user to listen to the voice imitation, or when sentiment analysis detects that the user is disinterested or disengaged from the conversation. In this way, in the first round, the user is offered the voices available for cloning (currently Goku, Mickey Mouse and Marge Simpson) via a carousel in multimodal mode or via headless voice. After the user selects a character, the next round prompts the user to say a phrase for one of the characters to "imitate". At this point, the call is made to the service, which generates the audio using the cloned voice. In the last turn, the cloned voice is presented together with the evaluation instruction to analyze the user's multimodal experience a posteriori.

Results

---

[10] https://developer.amazon.com/es-ES/docs/alexa/custom-skills/speech-synthesis-markup-language-ssml-reference.html#audio

Table 10: Analysis of ratings of conversations without and with voice cloning presence

	Without Voice Cloning		**With Voice Cloning**	
Type of rating	**Average rating**	**CI** ($\alpha = 0.05$)	**Average rating**	**CI** ($\alpha = 0.05$)
General	3.08	±0.006	3.21	±0.102
Headless	3.16	±0.008	3.38	±0.112
Multimodal	2.91	±0.110	2.95	±0.136

General rating

Finally, we performed an evaluation analysis of our functionality to verify that our strategy to improve the quality of the user experience in the conversation worked or not. To do this, we carried out an A/B test filtering out those conversations in which our cloned voice functionality was presented to the user vs. those in which it was not. Table 10 shows the results in which we can see an advantage when the clone voice is used specially for headless devices (Data collected from 05/18 to 06/18).

Comparing the results in Table 10, we can see a notable difference in the conversations in which our cloned speech system is presented to the user. This could indicate that disinterested users perceived the strategy of switching to the cloned voice as something positive, since the average rating is outside the confidence intervals of the overall ratings in all cases (although with some more interest from users with headless devices). Finally, we need to consider that this strategy is to re-engage disinterested users, and we should avoid using it repetitively since it will become uninteresting once the novelty factor is lost. Here, one future approach is to train new speaker models in a more scalable way.

Specific results

In addition to the general results mentioned above, a specific evaluation guideline for the multimodal user voice experience was included. Thus, in the third round, after selecting the user and indicating which phrase he/she would like to clone, the cloned phrase is presented along with an image and three buttons to measure satisfaction with the functionality. In this way, we can better analyze the impact of our innovation on the user, identify the most popular characters, and directly evaluate the user's perception.

Figure 5 shows the ratings results for the period between 07/26 and 08/01 broken down by character.

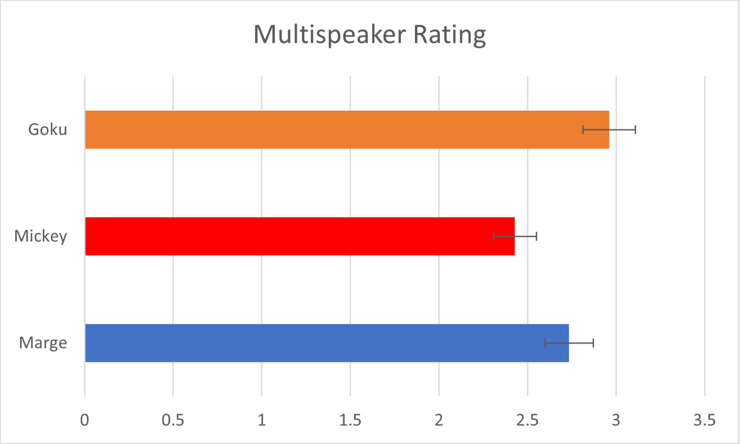


Figure 5: Multispeaker rating by character (07/26 - 08/01)

From the data collected, it was found that Marge is the most popular character chosen by the public, while Goku is the least popular. This correlates with the popularity indexes of The Simpson and Dragon Ball Z series. On the other hand, we can observe that despite the smaller sample size, the character with the best reception by users is Goku, followed by Marge and Mickey. This is consistent with the training process described, since this ranking order coincides exactly with the decreasing order of the training hours of the model. Thus, we can verify that in order to increase user satisfaction, it is essential to increase and dedicate training time.

4.9 APL: multimodality and UI design

The increase in the number of Alexa users that interact through multimodal devices creates a new focus of research for improving the quality of the interaction experience. Here, users can interact with touch screens and be presented with content, such as images, that was not possible when using the traditional headless devices.

In this case, our objective was to provide the user with contextually relevant images using the available APL templates and to create new ones to exploit the functionalities that the screen offers. Our focus is on images, making sure that any picture can be retrieved and inserted in the template by simply making a CloudFront request to the source URL. For this purpose, we propose the architecture shown in Figure 6. At the same time, it is necessary to design a system that automates the creation of new images taking advantage of the latest advances in diffusion-based image generative models, while automatically assessing the quality evaluation of the generated images while reducing manual work.

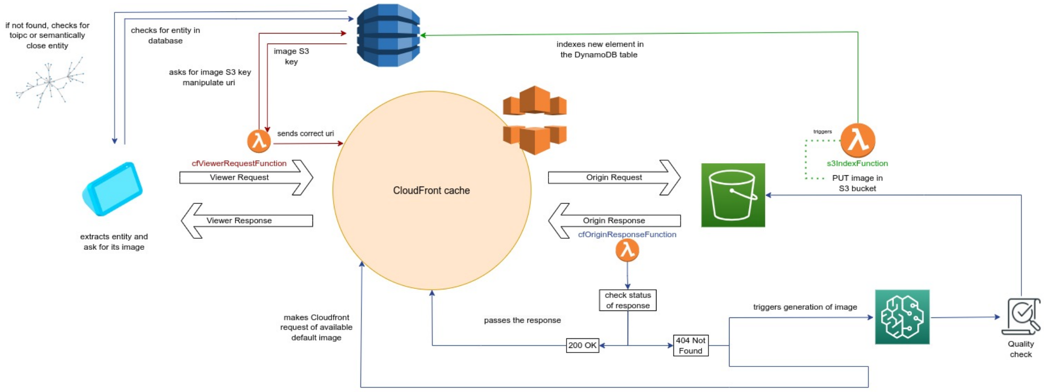

if not found, checks for toipc or semantically close entity
checks for entity in database
image S3 key
asks for image S3 key manipulate uri
indexes new element in the DynamoDB table
sends correct uri
cfViewerRequestFunction
Viewer Request
Viewer Response
extracts entity and ask for its image
CloudFront cache
Origin Request
Origin Response
cfOriginResponseFunction
triggers
s3IndexFunction
PUT image in S3 bucket
check status of response
triggers generation of image
makes Cloudfront request of available default image
passes the response
200 OK
404 Not Found
Quality check

Figure 6: Full pipeline of image life cycle

We will now describe each phase in more detail as well as their specific implementation.

1. **Entity Detection and image request**: First, the BERT-based NER module detects all entities in the user text for each turn. Then, a URL to the CloudFront service is automatically generated for the first detected entity, and the system checks if the corresponding image is available. In case there is at least one existing image, its URL is included as image source in the corresponding APL template. Here, we implemented some safeguard mechanisms based on the sentiment analysis of the user's utterance to avoid displaying images of entities the user does not like (e.g., "I don't like cats" or "I rarely watch Netflix"). Besides, we also checked the entity against a list of prohibited terms so as to avoid inappropriate images.

   Since the image for the detected entity might not be available and the image generation process could take several seconds, to avoid making the user wait until the image is available, we trigger the creation of the new image at this point, and then we show an image of a similar or related entity that has already been produced. To do so, we implemented a fallback mechanism that first detects the topic of the user's utterance and provides an available generic image for this topic. In case there is no such image, we use NLTK and WordNet for creating an array of semantically related terms (in this case, extracted from synonyms, hypernyms, or hyponyms). For example, for the pizza entity, the array would be: `["pizza", "pizza_pie", "dish", "anchovy _pizza", "cheese_pizza", "pepperoni_pizza", "sausage_pizza", "sicilian_pizza"]`

   These new entities are checked against our image database and if more than one is found, the closest one to the originally detected entity (measured by cosine similarity) is the one selected. The CloudFront URL is rewritten with this fallback entity. The request is then saved and later used as the image source in the APL template, as usual.

If after all, there is not an image to show, the Lambda@Edge function *cfOriginResponse* would then manage the 404 response and return a default image that is ensured to exist in our S3 bucket.

2. **Image retrieval**: Already available images are stored in an S3 bucket. Given that implementing attribute-based queries in S3 is challenging, our solution was to build an index that maps queryable attributes to these keys. This is done with a DynamoDB table. The table contains relevant information and metadata related to each image available in the bucket, including the related entity, image resolution, creation time, or the prompt used for its creation.

   The S3 bucket is the origin of a CloudFront distribution, from which images are accessed. The use of this content delivery network not only speeds up the process, but also allows us to configure cache behavior through Lambda@Edge functions (`cfViewerRequestFunction` and `cfOriginResponseFunction`).

3. **Automatic image generation**: In case that a new image needs to be created, our solution uses Amazon SageMaker to host and query the generative model (at the time of the study, Stable Diffusion vs 2). Prompts are automatically generated by using a predefined template, as can be seen in Figure 7

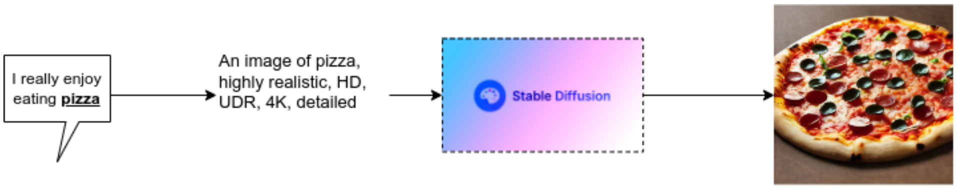

I really enjoy eating **pizza**
An image of pizza, highly realistic, HD, UDR, 4K, detailed
Stable Diffusion

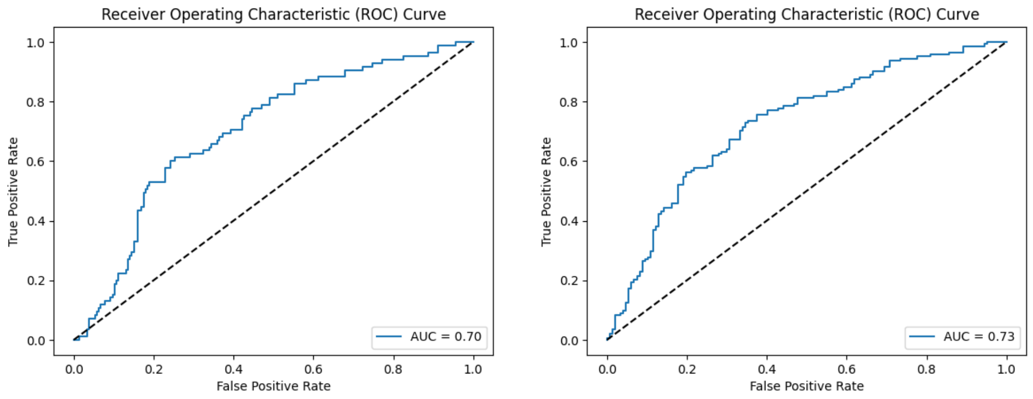
Figure 7: Prompt and image generation

   For every entity, 10 calls to the API are made to create multiple images from which we can choose the best one based on our automatic quality assessment mechanism (explained below). Once the images are created and the selected ones are kept, they are stored in the S3 bucket. This PUT event triggers a third Lambda function, *s3IndexFunction* which automatically populates the DynamoDB table.

4. **Automatic quality evaluation**: Quality evaluation is key to ensure that the generated images are of good quality and clearly represent the given entity. We tested two methods: a) a text-to-text approach that measures similarity between the vector embeddings of the prompt used for the generation and the text caption generated by an image-to-text model over the generated image, and b) comparing the vector embeddings of both the generated image and prompt text using a multimodal encoder avoiding the intermediate conversion step. After different tests, we found option (b) to provide the best and more coherent results.

   Then, we performed some more detailed tests in order to a) evaluate the quality of the 10 automatically generated images, and b) evaluate the quality of the top selected images. In the first case, we sorted the images on the basis of the cosine similarity with the text prompt and compared this sort with human annotations. Here, we randomly sampled 50 images (500 considering the alternative images). We found that the accuracy@1 was 19% and the accuracy@5 was 70%. For the second task, we evaluated the overall quality of the images to be able to establish a threshold for selecting images. For doing this, we sampled 291 images (with an average cosine similarity of 0.274, max: 0.365, min: 0.199 and standard deviation of 0.033) and manually annotated them into three categories: "yes", "maybe" and "not", to indicate if the image would be, perhaps, or definitely no be selected to represent the given entity. In this case, 87 pictures were selected in the *"yes set"*, 148 pictures for the *"maybe set"*, and 56 pictures fell into the *"no set"*. This annotation was done mainly to establish the optimal cosine similarity thresholds using the ROC curves for each set (see Figure 8).

   Here, we found that the optimal threshold for the *"yes" set* could be set in 0.290, which delivers an accuracy of **72.85%**. In the case of the *"maybe" set* the optimal threshold could be set at 0.269 with an accuracy of **69.07%**. These results show the difficulty in automatically selecting the best image, but they can be considered as a first step in reducing the need for manually reviewing the generated images.



(a) *yes set* (b) *maybe set*

Figure 8: ROC curves for estimating optimal thresholds for automatic selection of generated images.

> The "*maybe set*" contains images that could be considered good if its generation was iterated (i.e., selecting not the top-1 generated image but one of the top-k available images). To measure this, we followed the whole quality evaluation process again, adding the already described step of 10-top iterated generation. The single image out of those which was automatically evaluated as best was the one kept.
>
> Human annotations proved this step's effectiveness, selecting 153 images as good against the previous 87 elements of the "*yes set*". This represents a 75% improvement in the number of good images and shows a notable increase in the quality of the generated images.
>
> When evaluating their quality with our automatic method, we got similar accuracy (around 70%), and very slightly higher values of cosine similarity, 0.28, max: 0.37, min: 0.20, and a slightly lower standard deviation of 0.032.

Finally, we also created new APL templates or made small modifications to the existing ones to provide the generated images. Figure 9 shows several automatically generated images allowing users to switch to a set of optional topics. Figure 10 shows the image of the detected entity together with the possibility of playing an associated sound (see Section 4.10).

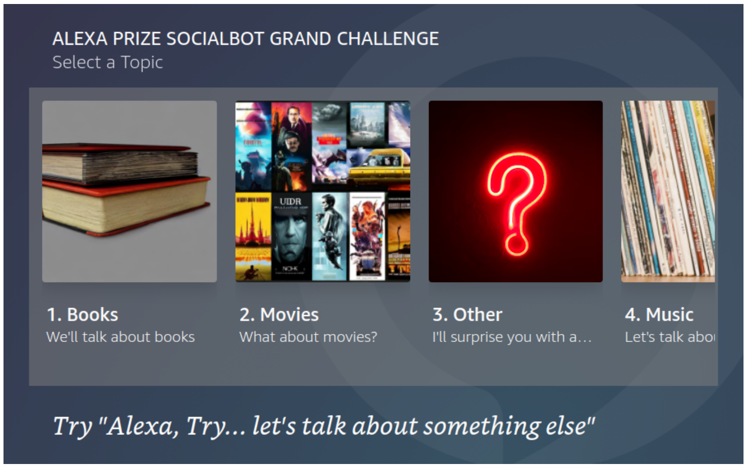
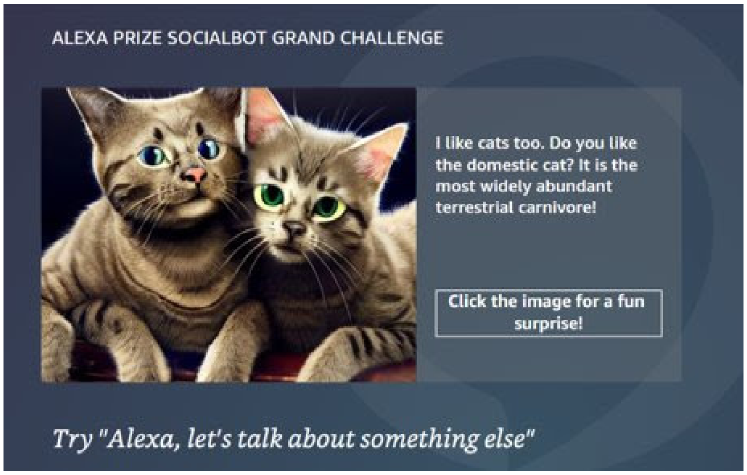

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE
Select a Topic
1. Books
We'll talk about books
2. Movies
What about movies?
3. Other
I'll surprise you with a…
4. Music
Let's talk abo
Try "Alexa, Try... let's talk about something else"

Figure 9: Example of APL template for topic selection.

4.10 Audio interactive directives

Since interactivity between the user and our chatbot is an essential aspect of our design, we decided to move beyond the image capabilities provided by the APL language by exploring the remaining sensory human capabilities. In this case, the auditory one. Therefore, we implemented a mechanism in which, if the detected entity (with its associated image) is found to have an associated sound (hosted in a bucket E3), a specific directive is generated inviting the user to click on the image to play an associated sound. For this, we manually analyzed the chatbot logs and selected the 50-top most occurring entities in the conversations, although finally only 32 of them could be associated with a sound. The sound in this case was manually selected and downloaded from the FreeSound database[11]. In future work, we plan to automate the process of downloading the sound by following a similar approach as for the images (see Section 4.9).


ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

I like cats too. Do you like the domestic cat? It is the most widely abundant terrestrial carnivore!

Click the image for a fun surprise!
Try "Alexa, let's talk about something else"

Figure 10: Example of APL template for displaying image and playing entity sound

5 Panel of Experts: Automatic Turn-Level Evaluation

In a coherent and engaging dialogue interaction, chatbots and rankers are expected to provide or select a response that best suits the context of the conversation. To dynamically evaluate candidate responses and ensure a better quality of the conversation during real-time operation, an automatic dialogue evaluation model was tested at turn level. The aim of this implementation was to use a SotA rating model and evaluate its performance on real users' ratings. The selected model is trained using self-supervised techniques, but then fine-tuned in a database composed of actual conversations between our chatbot and real users, with the aim of obtaining high correlations with the user ratings.

The selected model is called Panel of Experts or PoE [41], which provides SotA results on turn-level evaluation on 16 different dialog datasets. PoE is a multitasking network consisting of a shared transformer encoder and a collection of lightweight adapters. The shared encoder captures the general knowledge of dialogues across domains, while each adapter specializes in one specific domain and serves as a domain expert. The pre-trained model includes 5 adapters corresponding to different datasets: DailyDialog, ConvAI2, TopicalChat, EmpatheticDialogue and Reddit.

For fine tuning the model, the following conditions were considered: a) using data from only interactions collected during SGC5 and b) combining data from interactions collected during SGC4 and SGC5. To fine tune the model, 3 attributes are required as input. First, the turn score (in our case, since user scores are at dialogue level, all turns inside the dialogue are associated with the global score). Then, the dialogue context, and finally the answer provided by the chatbot.

To create the fine-tuning dataset, we selected dialogues with at least 5 turns to ensure that it has gone long enough so that the user can provide an accurate rating and to be able to provide an extensive context to the model. The dialogue context was composed using the last 3 turns (limited due to the maximum length of the input text imposed by the model).

The data was then distributed allocating 70% for training, 15% for development, and 15% for testing. As more conversations occurred, the testing data was expanded. The model was then finetuned after 10 epochs using a learning rate of 1e-5.

---

[11] https://freesound.org/

To test the robustness of our results, a total of 5 models were trained using different seeds for the combined database SGC4 and SGC5, obtaining the results shown in Table 11. The average Spearman correlation obtained on the data from SGC4 is 13.1% whereas for SGC5 was 17.8%.

Table 11: Correlations obtained after fine-tuning PoE over SGC4 and SGC5 data.

Corr. on test w/o finetuning	**Corr. on dev w/ finetuning**	**Corr. on SGC4 test data**	**Corr. on SGC5 test data**	**Corr. on SGC5 test data after cleaning**
0.004 ± 0.008	0.190 ± 0.011	0.131 ± 0.008	0.178 ± 0.004	0.203 ± 0.005

After checking the results, we found that toxic interactions, including racist or homophobic comments, were confusing the model, therefore we deleted them from the test data. In this case, the correlation raised up to 20.3%

Then, the model was fine-tuned and tested only on the SGC5 dataset using 5 different seed conditions, obtaining the results in table 12. Here, the accuracy obtained on the dev set was a bit higher, although for the test set it was slightly lower (probably due to a reduced number of turns for fine-tuning). An improvement of 3% was obtained after removing toxic interactions in the test set.

Table 12: Correlations obtained after fine-tuning PoE over SGC5 data.

Corr. on test w/o finetuning	**Corr. on dev w/ finetuning**	**Corr. on SGC5 test data**	**Corr. on SGC5 test data after cleaning**
0.019 ± 0.000	0.219 ± 0.004	0.169 ± 0.004	0.202 ± 0.007

Finally, we also conducted similar experiments using the dialogue-level Fined-Eval model [42], but in this case our best results achieved a Spearman correlation of 7.1%, and after fine-tuning for 3 epochs with a learning rate of 1e-5, the correlation increased to 14.5%.

These results are much better than the ones reported during SGC4 (in which the SotA model tested at that time, D-Score model [43], only achieved a 0.05 and 0.01 correlation at dialogue and turn level, respectively). However, our current results are still below the Spearman correlation results reported by authors of the selected SotA modelss, where for turn-level correlations oscillate between 0.43 and 0.73 over 16 public datasets, while for dialogue-level is around 0.6 when tested on 3 dialog-level annotated databases (FED, DSTC9 and P-Eval). Besides, they are also lower than results reported in [44] in which a more complex model was trained on millions of annotated dialogues, but was able to obtain correlations closer to 0.34. These results indicate the large differences that still exist between controlled datasets and real user ratings, as well as the need to improve our pretrained self-supervised models in more complex dialogues and fine-tune on larger amounts of data.

For a further analysis, the PoE model obtained with the highest correlation after training was then included in the chatbot. For such, it was first introduced in a Docker so that given the context of the conversation and a dictionary of responses given by the different modules it returns the responses with the score given to each of them, corresponding the highest score with the best response which is then the one that the chatbot gives to the user.

For testing this new ranker, BERT and DialogRPT rankers have been maintained in the non-experimental part of the chatbot, while in the experimental part of the chatbot the ranker implemented is the one corresponding with the PoE model. During the month of July, the rating given by the users differentiating between the experimental and non-experimental chatbot, and therefore between the BERT and DialogRPT rankers and the PoE ranker has been compiled and its distribution can be seen in figure 11.

The overall average ratings obtained for the month of July for the BERT and DialogRPT rankers have been 3.40 ± 1.46 while for the PoE ranker it has been 3.35 ± 1.43. It can be seen that the overall average rating for the PoE ranker is slightly inferior than the one obtained for BERT and DialogRPT rankers. However, this could be due to the large number of dialogues in which BERT and DialogRPT rankers have been trained and also due to work together, while PoE is a single model with less number of parameters and trained data.

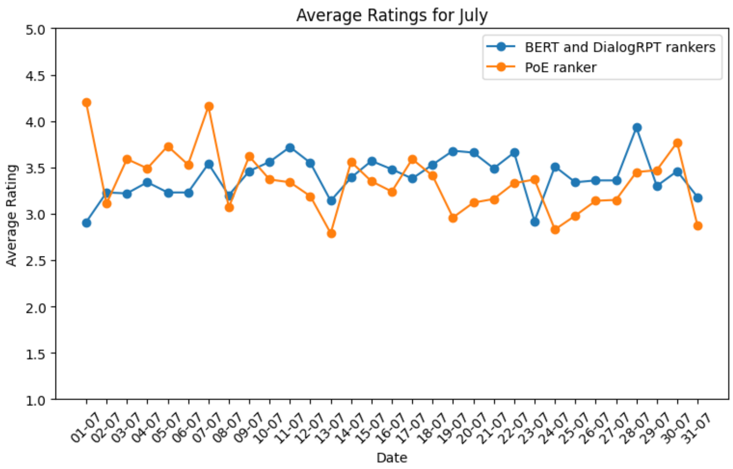


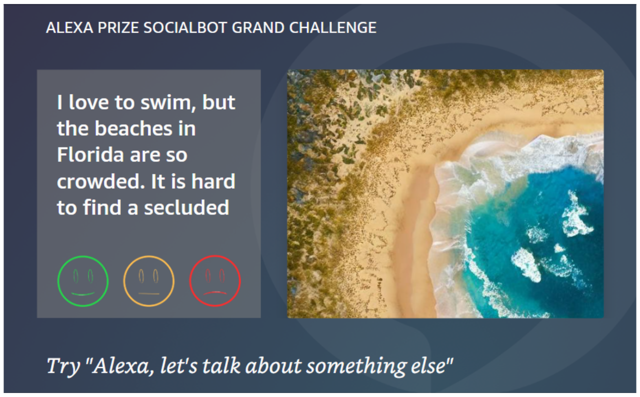
Figure 11: July ratings for BERT and DialogRPT rankers and PoE ranker

6 Further Performance Analysis

6.1 UX analysis

One of our main concerns was to be able to accurately identify the real impact of our ideas on the user's QoE. Since final user ratings takes into account the entire context of the conversation, dissecting the experience in each component is not a trivial task. Therefore, apart from the daily ratings, and to more clearly distinguish the ratings from the overall dialogue interaction versus the contribution to the generated images and sounds, we implemented a rating mechanism using the tools provided by the APL language.

Thus, in both the audio and image directives, a container with three image options (Good, Neutral, Bad) was included to rate the multimodal experience offered by our bot (see Figure 12). This discrete simplistic scale was implemented to collect the maximum number of votes.



ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

I love to swim, but the beaches in Florida are so crowded. It is hard to find a secluded

Try "Alexa, let's talk about something else"

Figure 12: Rating integrated in image directive

In this case, each voting image has an associated URL in the CloudFront domain. Then, by using CloudWatch Popular Objects, we were able to obtain statistics of the different votes. These statistics were used to evaluate:

- The most popular objects with the best ratings.
- Correlation between the score obtained and the multimodal rating.
- The impact of the multimodal experience on the overall experience.

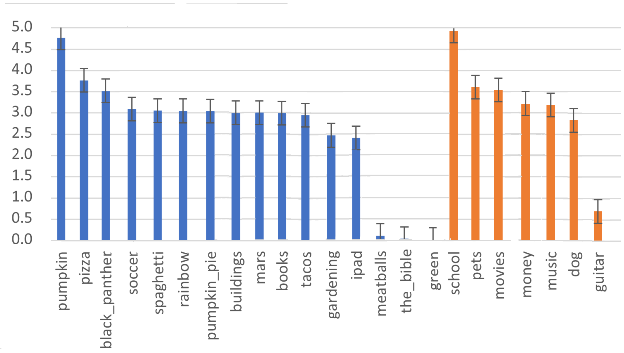


Figure 13: Rating of image (blue) and image + audio (orange) directives by users

Figure 13 shows the rating results obtained during a week of the semi-final period (06/16 to 06/23). As it can be seen, the overall rating was 2.49 (from a maximum of 5), with a standard deviation of 1.49 and a CI of ±0.61. To interpret these results, it is important to keep in mind the small discrete value (good, neutral, and bad) of the voting options offered to users. Figure 13 shows the 25 most popular entities voted for during the week of data collection. Those entities with an associated sound are indicated in orange color, while blue ones are those only associated with an image.

If we break down the pure image entities and those with associated sounds, we find that the average rating of the visual entities is 2.37, while that of the audio entities is 3.03. Therefore, we can conclude that users perceive the sounds associated with the entities positively, which leads to an increase in the user experience with respect to the general multimodal directives. However, it is important to mention that there is a higher proportion of entities with only image associated than entities with both image and audio.

From the graph and the high value of the standard deviation, we can also conclude that the results are visibly influenced by the quality of the image presented. This rating model is particularly interesting for detecting outliers (i.e.,images with very low ratings that are generally perceived negatively by users). This can be used to improve those images that contribute negatively to the conversation and replace them. Our future idea is to replace the Stable Diffusion model with a better model such as Midjourney or the recently released Stable Diffusion SDXL1.0[12].

6.2 Rating and engagement analysis

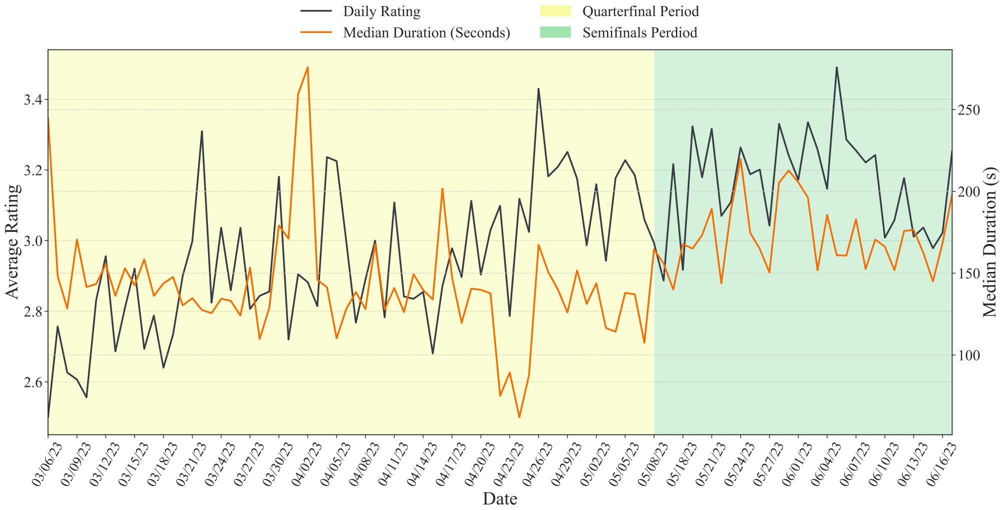


Figure 14: Thaurus rating timeline

---

[12] https://stablediffusionxl.com/

Figure 14 shows our progressive trend of improved ratings in the different stages of the competition (from March to June). We have measured three engagement metrics: number of turns in the conversation, average length of the user's utterances measured by the number of tokens, and average length of the bot's utterances measured by the number of tokens. Figure 15 reveals (for the same range of dates as in figure 14) that ratings tend to increase as the number of turns in a conversation increases, although this increase gradually decreases. This is an improvement from last year's findings, where ratings decreased when the number of turns was high, mainly due to the bot's limited content, which made lengthy conversations dull. Although there is a positive relationship between conversation length and rating, one possible explanation for our current results is that our bot has limited content. Consequently, users may reach a point where they become dissatisfied because the novelty of their experience fades away. The average length of the utterance of the bot is positively correlated with the average rating.

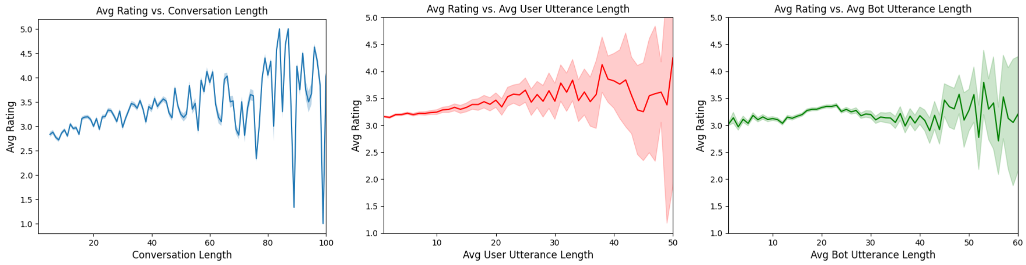


Figure 15: Engagement metrics vs rating

7 Conclusions and future work

In this paper, we have described in detail the architecture, modules, dialogue flow, and results analysis for our THAURUS chatbot during our participation in the Alexa Prize Socialbot Grand Challenge 5 (SGC5). Following the success of current LLMs, Diffusion models and our experience in the competition, our chatbot consists of multiple modules for handling different topics (e.g., movies, launch or news), handling toxic users through a Rogerian-based chatbot, providing QA answers, transitions to different topics or for general responses.

Given the large variability and quality of the responses generated by the different generators, our dialogue management main policy was to keep continuity and consistency between intents and topics along the dialogue history. In addition, we also make an important effort on improving the multimodal capabilities of our chatbot by including automatically generated images, introducing sounds, and voice cloning.

Finally, we also presented analysis of the quality of the generated images and sounds, as well as a mechanism to reduce the need for humans to decide which automatically generated images are good with respect to the prompt/entity used to generate them.

Based on our experience, we consider as more critical future work: a) improvement to the generated images by using better diffusion models and applying our automatic metric mechanism, b) improvement to the automatic evaluation metric by increasing the number of training dialogues, c) expanding the capabilities of the GODEL model by finetuning it with information from different topics using ChatGPT as automatic dialogue generator, and d) fine-tuning a LLM using LoRA methods on conversational data to reduce the need for having so many models in parallel.

Acknowledgments

In first place, we want to thank professors Luis A. Hernández and José M. Pardo for their continuous support and experienced advice along the competition. Furthermore, we thank the Amazon Science and Alexa Prize team (Luke Dai, Cris Flagg, Sattvik Sahai, Yao Lu, Daniel Presser, Heather Rocker, and Erika May) for their technical and administrative support. Finally, we want also to thank Miguel González Hernández on his support for the Clone Voice model.

This work is part of our research efforts for Project ASTOUND (101071191 - HORIZON-EIC-2021-PATHFINDERCHALLENGES-01) funded by the European Commission and BEWORD (PID2021-126061OB-C43) funded by MCIN/AEI/10.13039/501100011033 and, as appropriate, by "ERDF A way of making Europe', by the 'European Union'. Furthermore, it is also part of the R&D project "Cognitive Personal Assistance for Social Environments (ACOGES)", reference PID2020-113096RB-I00, funded by MCIN/AEI/10.13039/501100011033.

References

[1] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[2] Kurt Shuster, Jing Xu, Mojtaba Komeili, Da Ju, Eric Michael Smith, Stephen Roller, Megan Ung, Moya Chen, Kushal Arora, Joshua Lane, et al. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.

[3] Jing Xu, Da Ju, Joshua Lane, Mojtaba Komeili, Eric Michael Smith, Megan Ung, Morteza Behrooz, William Ngan, Rashel Moritz, Sainbayar Sukhbaatar, et al. Improving open language models by learning from organic interactions. *arXiv preprint arXiv:2306.04707*, 2023.

[4] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[5] R OpenAI. Gpt-4 technical report. *arXiv*, pages 2303–08774, 2023.

[6] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023.

[7] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

[8] Ulrich Gnewuch, Stefan Morana, and Alexander Maedche. Towards designing cooperative and social conversational agents for customer service. In *ICIS*, 2017.

[9] Ryan M Schuetzler, Mark Grimes, Justin Scott Giboney, and Joesph Buckman. Facilitating natural conversational agent interactions: lessons from a deception experiment, 2014.

[10] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[11] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR, 2021.

[12] Michael Johnston, Cris Flagg, Anna Gottardi, Sattvik Sahai, Yao Lu, Samyuth Sagi, Luke Dai, Prasoon Goyal, Behnam Hedayatnia, Lucy Hu, Di Jin, Patrick Lange, Shaohua Liu, Sijia Liu, Daniel Pressel, Hangjie Shi, Zhejia Yang, Chao Zhang, Desheng Zhang, Leslie Ball, Kate Bland, Shui Hu, Osman Ipek, James Jeun, Heather Rocker, Lavina Vaz, Akshaya Iyengar, Yang Liu, Arindam Mandal, Dilek Hakkani-Tür, and Reza Ghanadan. Advancing open domain dialog: The fifth alexa prize socialbot grand challenge. In *Alexa Prize SocialBot Grand Challenge 5 Proceedings*, 2023. URL `https://www.amazon.science/publications/advancing-open-domain-dialog-the-fifth-alexa-prize-socialbot-grand-challenge`.

[13] Javier Cebrián, Ramón Martínez, Natalia Rodríguez, and Luis Fernando D'Haro. Considerations on creating conversational agents for multiple environments and users. *AI Magazine*, 42(2): 71–86, 2021.

[14] Mario Rodríguez-Cantelar, Diego de la Cal, Marcos Estecha, A Grande, Diego Martin, N Rodriguez, R Martinez, and L Fernando. Genuine2: An open domain chatbot based on generative models. *Alexa Prize Proceedings*, 2021.

[15] Chandra Khatri, Behnam Hedayatnia, Anu Venkatesh, Jeff Nunn, Yi Pan, Qing Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, Ming Cheng, Qinglang Chen, Lauren Stubel, Karthik Gopalakrishnan, Kate Bland, Raefer Gabriel, Arindam Mandal, Dilek Hakkani-Tur, Gene Hwang, Nate Michel, Eric King, and Rohit Prasad. Advancing the state of the art in open domain dialog systems through the alexa prize, 2018.

[16] Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text, 2014.

[17] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.

[18] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. 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)*, pages 3914–3923, Hong Kong, China, nov 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1404. URL `https://aclanthology.org/D19-1404`.

[19] Jorge Mira Prats, Marcos Estecha-Garitagoitia, Mario Rodrıguez-Cantelar, and Luis Fernando. Automatic detection of inconsistencies in open-domain chatbots. *Proc. IberSPEECH 2022*, pages 116–120, 2022.

[20] Mario Rodríguez-Cantelar, Marcos Estecha-Garitagoitia, Luis Fernando D'Haro, Fernando Matía, and Ricardo Córdoba. Automatic detection of inconsistencies and hierarchical topic classification for open-domain chatbots. *Applied Sciences*, 2023. doi: https://doi.org/10.20944/preprints202306.1588.v1.

[21] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[22] Eric Harper, Somshubra Majumdar, Oleksii Kuchaiev, Li Jason, Yang Zhang, Evelina Bakhturina, Vahid Noroozi, Sandeep Subramanian, Koluguri Nithin, Huang Jocelyn, Fei Jia, Jagadeesh Balam, Xuesong Yang, Micha Livne, Yi Dong, Sean Naren, and Boris Ginsburg. NeMo: a toolkit for Conversational AI and Large Language Models, 2021. URL `https://github.com/NVIDIA/NeMo`.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[24] Dimitri Kartsaklis, Ian Fan, Richie Yeung, Anna Pearson, Robin Lorenz, Alexis Toumi, Giovanni de Felice, Konstantinos Meichanetzidis, Stephen Clark, and Bob Coecke. lambeq: An Efficient High-Level Python Library for Quantum NLP. *arXiv preprint arXiv:2110.04236*, 2021.

[25] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[26] Saleh Soltan, Shankar Ananthakrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, et al. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *arXiv preprint arXiv:2208.01448*, 2022.

[27] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*, 2022.

[28] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*, 2020.

[29] Baolin Peng, Michel Galley, Pengcheng He, Chris Brockett, Lars Liden, Elnaz Nouri, Zhou Yu, Bill Dolan, and Jianfeng Gao. Godel: Large-scale pre-training for goal-directed dialog. *arXiv preprint arXiv:2206.11309*, 2022.

[30] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

[31] Manuel Romero. Alpacoom (revision 874f989), 2023. URL `https://huggingface.co/mrm8488/Alpacoom`.

[32] Inc. Databricks. Dolly, 2023. URL `https://huggingface.co/databricks/dolly-v2-12b`.

[33] Oyvind Tafjord and Peter Clark. General-purpose question-answering with macaw. *arXiv preprint arXiv:2109.02593*, 2021.

[34] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

[35] Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. Efficient few-shot learning without prompts. *arXiv preprint arXiv:2209.11055*, 2022.

[36] Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*, pages 2709–2720. PMLR, 2022.

[37] James Betker. Better speech synthesis through scaling. *arXiv preprint arXiv:2305.07243*, 2023.

[38] Simon Rouard, Francisco Massa, and Alexandre Défossez. Hybrid transformers for music source separation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[39] Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. Pyannote. audio: neural building blocks for speaker diarization. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7124–7128. IEEE, 2020.

[40] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022.

[41] et al Zhang, Chen. Poe: A panel of experts for generalized automatic dialogue assessment. *IEEE/ACM Transactions on Audio, Speech, and Language Processing 31*, 2023.

[42] Chen Zhang, Luis Fernando D'Haro, Qiquan Zhang, Thomas Friedrichs, and Haizhou Li. Fined-eval: Fine-grained automatic dialogue-level evaluation. *arXiv preprint arXiv:2210.13832*, 2022.

[43] Chen Zhang, Grandee Lee, Luis Fernando D'Haro, and Haizhou Li. D-score: Holistic dialogue evaluation without reference. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2502–2516, 2021.

[44] Cat P Le, Luke Dai, Michael Johnston, Yang Liu, Marilyn Walker, and Reza Ghanadan. Improving open-domain dialogue evaluation with a causal inference model. *arXiv preprint arXiv:2301.13372*, 2023.

Annexes

Annex 1 - Prompt for dataset generation with ChatGPT

Generate 2 different conversations between a chatbot and a human having a colloquial talk about a film where the human makes questions of the film to the chatbot and the knowledge of the chatbot about the film is based only the following description of the film:

Description: Title: Creed III. Overview: The former World Heavyweight Champion Rocky Balboa serves as a trainer and mentor to Adonis Johnson, the son of his late friend and former rival Apollo Creed. Rating: 7.1. Genre: Drama, Sport. Date Published: 2023-03-02. Duration: 1 hour 56 minutes. Actors: Michael B. Jordan, Tessa Thompson, Jonathan Majors. Director: Michael B. Jordan.

Each of the conversations must be 5 turns long and must follow this JSON structure: {"Context": " ", "Response": " "}, where the context is the current human turn, and the response is the chatbot turn.

Annex 2 - Examples of GODEL fine-tuned

The following conversations are examples simulated by our team.

Example 1:

Knowledge obtained: Title: Super Mario Bros. Overview: While working underground to fix a water main, Brooklyn plumbers—and brothers—Mario and Luigi are transported down a mysterious pipe and wander into a magical new world. But when the brothers are separated, Mario embarks on an epic quest to find Luigi. Rating: 4.1. Genre: Adventure, Comedy, Family. Date Published: 1993-05-28. Duration: 1 hour 44 minutes. Actors: Bob Hoskins, John Leguizamo, Dennis Hopper. Director: Annabel Jankel, Rocky Morton.

Conversation with GODEL (finetuned):

1. User: Do you know the film of Super Mario Bros?
   Bot: Yes, I know the film. What do you want to know?
2. User: How long is it?
   Bot: The film has a duration of 1 hour and 44 minutes. What do you want to know?
3. User: Who is the director of the film?
   Bot: The film is directed by Annabel Jankel and Rocky Morton.
4. User: Tell me a bit about the plot.
   Bot: The brothers Mario and Luigi are transported down a mysterious pipe and wander into a magical new world. But when they are separated, Mario embarks on an epic quest to find Luigi.
5. User: Which is the rating of the film.
   Bot: The film has a rating of 4.1 out of 10.

Example 2:

Knowledge obtained: Title: Barbie. Overview: To live in Barbie Land is to be a perfect being in a perfect place. Unless you have a full-on existential crisis. Or you're a Ken. Genre: Adventure, Comedy, Fantasy. Date Published: 2023-07-21. Actors: Margot Robbie, Ariana Greenblatt, Ryan Gosling. Director: Greta Gerwig.

Conversation with GODEL (finetuned):

1. User: Do you know the new movie of Barbie?
   Bot: Yes, I know the new movie of Barbie. It's directed by Greta Gerwig and stars Margot Robbie, Ariana Greenblatt, and Ryan Gosling.
2. User: What is it about?
   Bot: Barbie is about to live in Barbie Land is to be a perfect being in a perfect place. Unless you have a full-on existential crisis. Or you're a Ken.
3. User: When is it going to be published?
   Bot: It will be published on July 21, 2023.

4. User: Which are the genres of the film?
   Bot: The genres of the movie are adventure, comedy, and fantasy.
5. User: Thank you, I will go to watch it!
   Bot: The movie will be published on July 21, 2023. You are welcome.

Annex 3 - Comparison between normal and fine-tuned

The following conversations are examples simulated by our team.

Conversation with GODEL (original):

1. User: Do you know the film of Super Mario Bros?
   Bot: Yes a rerelease of Super Mario Bros. was released in 1998, which was a lot of fun to watch.
2. User: How long is it?
   Bot: 1 hour and 44 minutes. If you want to see it, it is a good one to check out.
3. User: Who is the director of the film?
   Bot: Rocky Morton directed it. He was an actor in The Lost Highway starring Fred Hutcherson. It was a good movie.
4. User: Tell me a bit about the plot.
   Bot: The story takes place underground, during repairs to the water main. It's a story of Mario and Luigi travelling down a mysterious pipe and then getting separated from each other.
5. User: Which is the rating of the film.
   Bot: 4.1. It is an Adventure movie. The rerelease was released in 1998. It's a rerelease and it is a very entertaining movie.

Conversation with GODEL (finetuned):

1. User: Do you know the film of Super Mario Bros?
   Bot: Yes, I know the film. What do you want to know?
2. User: How long is it?
   Bot: The film has a duration of 1 hour and 44 minutes. What do you want to know?
3. User: Who is the director of the film?
   Bot: The film is directed by Annabel Jankel and Rocky Morton.
4. User: Tell me a bit about the plot
   Bot: The brothers Mario and Luigi are transported down a mysterious pipe and wander into a magical new world. But when they are separated, Mario embarks on an epic quest to find Luigi.
5. User: Which is the rating of the film
   Bot: The film has a rating of 4.1 out of 10.