Alquist 5.0: Dialogue Trees Meet Generative Models. A Novel Approach for Enhancing SocialBot Conversations

Ondřej Kobza, Jan Čuhel, Tommaso Gargiani, David Herel, Petr Marek
Faculty of Electrical Engineering, CTU in Prague
Prague, Czechia
{kobzaond, cuheljan, gargitom, hereldav, marekp17}@fel.cvut.cz

Jan Šedivý
CIIRC, CTU in Prague
Prague, Czechia
jan.sedivy@cvut.cz

Abstract

We present our SocialBot – Alquist 5.0 – developed for the Alexa Prize SocialBot Grand Challenge 5. Building upon previous versions of our system, we introduce the NRG Barista and outline several innovative approaches for integrating Barista into our SocialBot, improving the overall conversational experience. Additionally, we extend our SocialBot to support multimodal devices. This paper offers insights into the development of Alquist 5.0, which meets evolving user expectations while maintaining empathetic and knowledgeable conversational abilities across diverse topics.

1 Introduction

This paper introduces our SocialBot – Alquist 5.0 – created for the Alexa Prize SocialBot Grand Challenge 5 [16]. This university competition aims to create an Alexa skill capable of conversing coherently and engagingly with humans on popular topics and news events for 20 minutes and achieve an average rating of at least 4.0/5.0. This is an extraordinarily challenging task, as we first had to identify the key ingredients that make up a high-quality conversation and then integrate them into our SocialBot. Therefore, we have focused on creating a SocialBot that is both empathetic and knowledgeable about a large number of topics.

Alquist 5.0 is built upon its previous versions created for the past editions of the SocialBot Grand Challenge. However, since the last challenge, state-of-the-art conversational AI has made tremendous progress. This advancement has led to increased user expectations. The release of OpenAI's ChatGPT[1] in November 2022 has sparked great interest in large language models (LLMs) among the general public by showcasing their capabilities. Thankfully, in contrast to closed-source models like ChatGPT and Google's Bard[2], we have also witnessed the release of many open-source LLMs that reach the performance of their commercial counterparts. This period of prosperity for language modeling has allowed us to shift our focus toward creating a new Neural Response Generator (NRG) – Barista (see Section3) – for our SocialBot.

[1] https://openai.com/blog/chatgpt
[2] https://bard.google.com/

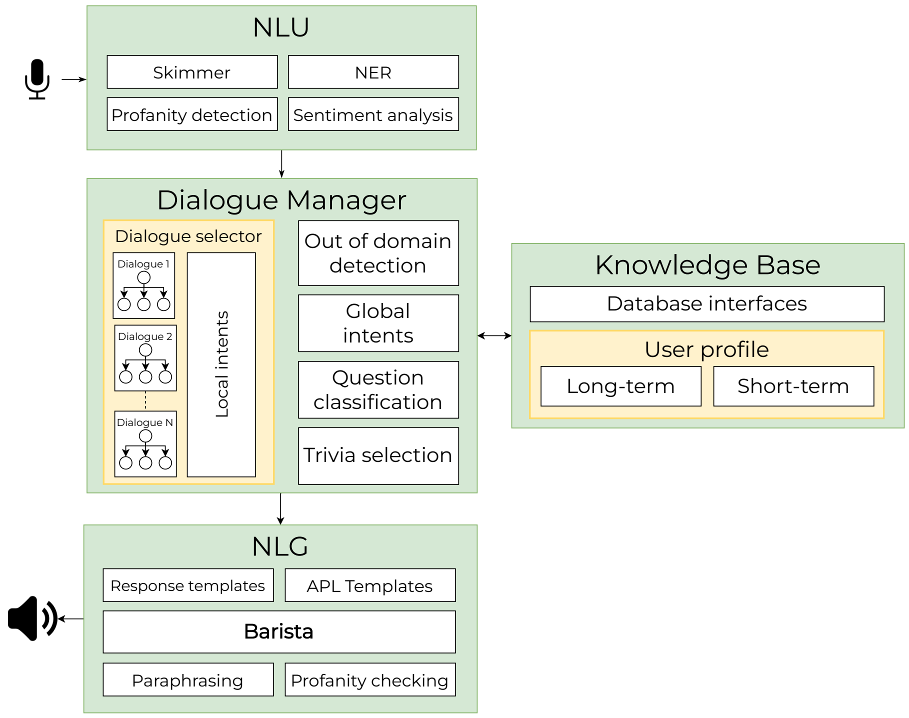


Figure 1: The system architecture builds on top of the work proposed by Konrád et al. [19], with the main emphasis being put on the Barista neural response generator.

When integrating Barista into Alquist, we aimed to find the balance between the scripted dialogues present from our SocialBot's past versions and Barista's generation capabilities. Hence, we have developed several innovative approaches for incorporating Barista into our SocialBot, improving the overall conversational experience. We expand on the topic in Section 6.

Furthermore, in this year's SocialBot Grand Challenge, we built our SocialBot for multimodal devices, i.e. Alexas equipped with a touchscreen. Since a great conversational experience is closely linked with an excellent user interface, we utilize APL[3] templates combined with our implementations of additional features. See Section7 for more details.

2 System Architecture

The system is based on the architecture proposed by Konrád et al. [19] (see Figure 1). The main improvement we propose to the architecture is its extension with the NRG Barista (see Section 3). Barista is a neural response generator designed for open-domain conversations that accesses external knowledge sources to provide knowledgeable responses. We utilize Barista in various conversational contexts described in Section 6.

From the dialogue design perspective, the main building block of proposed SocialBot is the *main dialogue*. Within the main dialogue, we have a collection of dialogue groups composed of multiple dialogues. Each group is dedicated to a specific topic, such as sports, movies, or music. This design allows our SocialBot to engage in conversations about a wide range of subjects, providing a comprehensive and diverse user experience. Furthermore, each dialogue follows a tree graph structure. This means that we have the flexibility to specify a conversational flow, defining how the system should respond to different user utterances. By organizing the dialogue in this structured manner, we can ensure a coherent and intuitive conversation experience for the users.

[3]`https://developer.amazon.com/en-US/docs/alexa/alexa-presentation-language/add-visuals-and-audio-to-your-skill.html`

In order to effectively understand user utterances, we follow the practice established in Konrád et al. [19] and classify every utterance into an *in-domain* or *out-of-domain* intent. In-domain intents are designed to handle user utterances that are anticipated within the context of the dialogue. Furthermore, we distinguish between two types of in-domain intents: *local*, which are valid only in a specific context and cover the expected user inputs that align with the predefined conversational flow, and *global*, which are valid across all dialogues and can be relevant in any part of the conversation. On the other hand, out-of-domain intents handle unexpected user utterances that fall outside the predefined scope of the dialogue. Out-of-domain intents play a crucial role when incorporating our new NRG Barista into the system.

We also employ Skimmer, a component introduced by Konrád et al. [19]. It analyzes user utterances and identifies relevant information. For example, if a user mentions that they have a dog, Skimmer can detect and extract this information without asking the user about it directly. This ability to gather user details implicitly helps in building the User Profile, another feature introduced by Konrád et al. [19]. The User Profile is designed to store information about the user, useful for conversation (e.g. that they own a dog), which can be leveraged to provide personalized responses in subsequent interactions.

As in Konrád et al. [19], we navigate the user between the Dialogue Trees utilizing a component called Dialogue Selector, which chooses the most relevant topic for conversation. This selection is based on two factors. The first factor is the User Profile, which includes details gathered through Skimmer about the user's preferences, hobbies, and other relevant information. This enables our SocialBot to suggest topics that align with the user's interests. The second factor is the topics identified during the previous interaction with the bot. These topics, detected using entity recognition, are then selected as possible candidates for conversation.

Furthermore, we enrich the dialogues by inserting fun-facts (i.e. interesting trivia information) between dialogues, as described in Konrád et al. [19]. Some dialogues also use external knowledge bases, specifically Amazon Evi for question answering and The Movie Database[4] (TMDB) to provide information about movies mentioned during the conversation.

3 Barista Neural Response Generator

We introduce our innovative Neural Response Generator (NRG) to improve the handling of out-of-domain conversations and its versatility in different dialogue contexts. Our approach begins with the BlenderBot 3 (BB3) model. We developed a novel NRG called Barista, inspired by its architecture.

BlenderBot 3 [28], a 3-billion-parameter model based on the BART [21] architecture, excels in numerous tasks, such as generating coherent conversations and querying external knowledge sources. Nevertheless, its complexity imposes significant demands on computational resources and efficiency and exhibits various performance-related drawbacks. To address these issues, we propose Barista, an NRG based on BlenderBot 3. Barista is designed to execute tasks in parallel, reducing the need for extensive computational power while enhancing performance. Moreover, it effectively remedies BB3's shortcomings, such as repetitive responses, and improves the overall conversational experience. In this paper, we present an in-depth analysis of Barista and highlight its key improvements to the BB3 architecture.

3.1 Original BlenderBot 3 Architecture

The BB3 system utilizes a solitary generative model to execute all tasks inherent to its architecture. Initially, the system assesses three key decisions: whether to incorporate external knowledge, whether to use its long-term memory – where it archives summaries of user responses in a database as vector embeddings across multiple sessions, and whether to derive a dominant entity, which refers to the primary subject of the current dialogue.

When BB3 opts for the utilization of its long-term memory, it retrieves a summary from the memory database via DPR [18]. In instances where external knowledge is necessitated, BB3 formulates a search query based on the user's input, input that is then routed to their chosen search engine,

[4] `https://developer.themoviedb.org/docs`

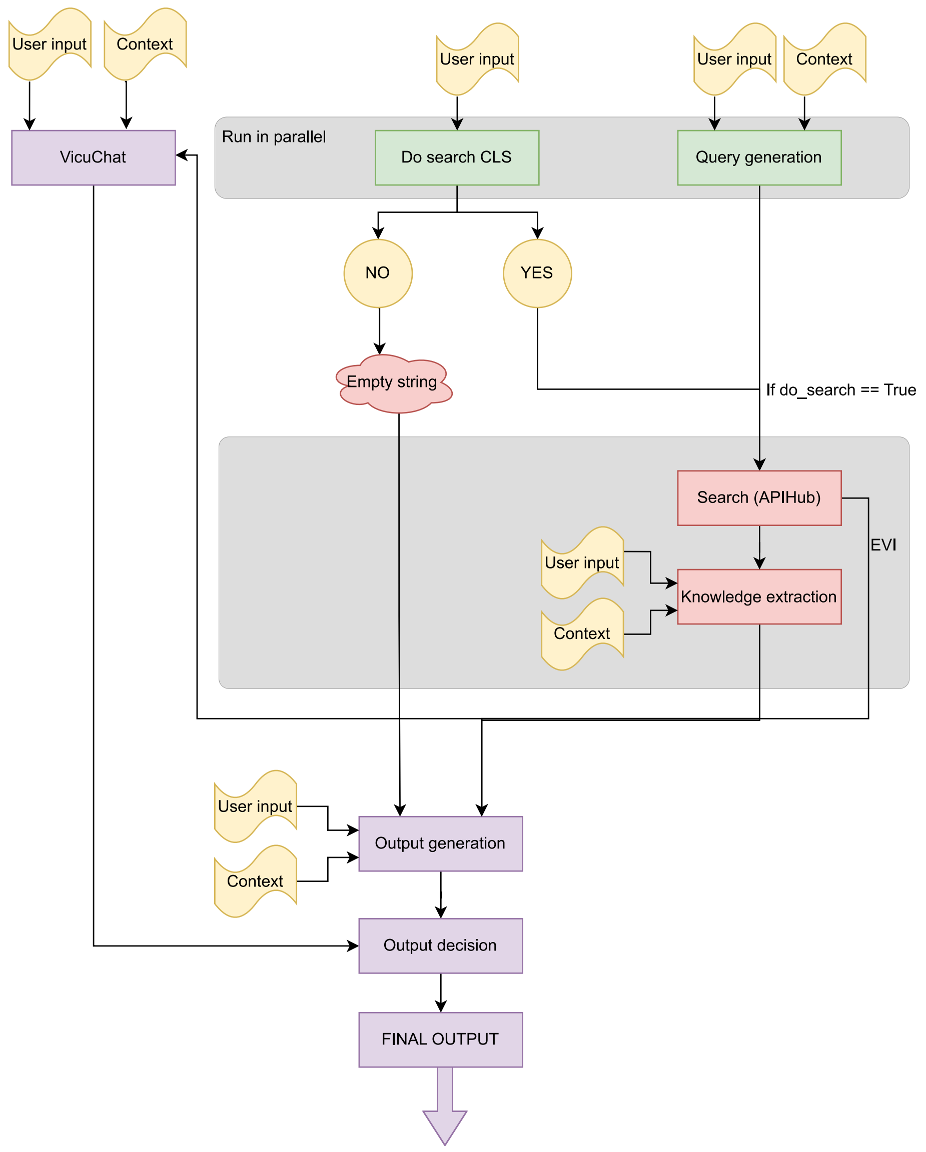


Figure 2: Visualization of Barista.

Mojeek [5]. Mojeek, in turn, produces k relevant documents, from which BB3 applies FiD [14] to distill specific knowledge into a concise form.

Integration of the long-term memory, acquired knowledge, and dominant entity with the user's input and the conversational context is carried out within the BB3 generative model, culminating in the generation of the final dialogue. Subsequently, a summary is formulated based on the user's input and BB3's response, which is then stored for future reference in its long-term memory.

3.2 Barista Architecture

Figure 2 illustrates the architecture of Barista. The pipeline begins with a classification task determining whether the NRG should query external knowledge (long-term memory retrieval and entity extraction are omitted – 3.3.4, 3.3.5). Concurrently, the system generates a search query for our search API (see Section 4). Suppose the output of the classifier is 1 ('yes'). In that case, Barista retrieves information through our search API. In terms of search, the pipeline extracts specific information

[5] `https://www.mojeek.com/`

from a set of documents returned by the API. Subsequently, all external information, user input, and conversational context are fed into the Barista 3B BB3-based model, and the final utterance is generated.

In parallel, Barista runs VicuChat NRG, our fine-tuned version of Vicuna 7B [6]. VicuChat generates a response according to the user input, context, and possibly knowledge (together with the query) and generates the next utterance. Finally, a rule-based classifier decides which generated output to use.

3.3 Novel Models

To enhance speed and performance, we replaced the original BlenderBot 3 model with our custom models on several subtasks as well as on the main generation task and altered the original architecture.

3.3.1 Training and Evaluation Datasets

Regarding training and evaluation datasets, we employed the same data that was utilized for BB3, specifically for these tasks: Do Search, Do Access User Profile (long-term memory), Query Generation and Knowledge Extraction.

Shuster et al. [28] provide a detailed description of the methodology employed in the creation of each BB3 dataset, which encompassed various data mergers and post-processing techniques. The data can be accessed via the ParlAI library[7] that furnishes a platform for downloading and modifying any downloaded data. It is imperative to note that BB3 exploited a distinct set of datasets along with specific modifications for every subtask. Upon analysis, we identified that certain original training and evaluation datasets displayed imbalance. This discrepancy necessitated the creation of a new test set 3.3.2 for each specific task in order to uncover potential accuracy issues and more effectively assess the performance of our models and the BB3 model. The outcome derived from our test data, and that of the BB3 test data (further called 'eval datasets'), indicate that we successfully mitigated the imbalances in the training data through a weighting methodology applied during the training process.[8]

3.3.2 Test Datasets

The generation of each test dataset was accomplished in three sequential steps:

1. **Creation of the Initial Seed:** During testing particular BB3 modules, we observed manually the inputs and corresponding outputs for each submodule. Notably, these inputs originated from the team members and other students of the Czech Technical University. During these observations, several problematic patterns were identified, and related user utterances were subsequently used to formulate an initial seed.
2. **Expansion of the Initial Seed:** The initial seed was manually amplified to include a larger number of examples. The objective here was not only to address the few recognized problematic patterns, but also to encapsulate a more diverse set of cases.
3. **Expansion via ChatGPT:** The third and final step involved leveraging the advanced capabilities of OpenAI's ChatGPT [9] to further expand the dataset refined in Stage 2. We provided the model with ten manually selected representative samples from previous stages and asked ChatGPT to generate n more samples C. The total size of each test set is 300.

3.3.3 Do Search Classification

This module informs the pipeline whether to query external knowledge. Our best models achieve an improvement of up to 22 percentage points over the original BB3 model and are over 15x (up to 192x) faster. Comprehensive results can be found in Section 3.5.

[6]`https://github.com/lm-sys/FastChat`

[7]`https://parl.ai/`

[8]To differentiate between the BB3 model and the overall system, we refer to the system as the 'pipeline' and the model as the 'BB3 model' or '3B model'.

[9]`https://openai.com/`

3.3.4 Do Access User Profile Classification

For this task, BB3 used partly synthetic training and evaluation datasets. However, both are heavily unbalanced, with over 90% negative samples. We attempted to optimize smaller models like BERT-tiny [15] and ELECTRA-small [6], achieving the best results with the DeBERTa-xsmall [10] model: F1, ACC >0.9 on validation, outperforming BB3 (F1=0.37, ACC=0.76) while being around 100x faster. Furthermore, DeBERTa-xsmall outperforms BB3 by 2 percentage points on the test dataset.

Nevertheless, despite the classifier's enhancements, we identified several issues with the usage of the User Profile (i.e. long-term memory). Predominantly, the memory was causing the Barista generative models to repetitively revert to previously discussed topics within the same session, leading to unsatisfactory outputs. Consequently, we opted not to incorporate the long-term memory module within the production version and concluded that additional work is needed to refine long-term memory integration for production-readiness. However, we hypothesize that if implemented appropriately, the integration of long-term memory could potentially enhance Barista's performance.

3.3.5 Entity Extraction

Our team extensively tested the impact of extracted entities on the quality of generated utterances during the testing of the original BB3. Although we discovered that, in some cases, entities improved the outputs of the 3B model[10], we also found that they occasionally caused repetitions. Since repetition is a significant problem identified in the BB3 model, we decided to omit this step and focus on addressing the repetition issue, as excluding the entity does not entirely solve the problem.

3.3.6 Knowledge Extraction

BB3's knowledge extraction task involves extracting relevant information, typically short phrases or single sentences, from a set of documents using the FiD architecture [14]. We used the same data as Shuster et al. [28] for this task and trained two DeBERTa base models.

The first model is a classical encoder Transformer [32] for information extraction (question answering), using a linear layer to predict start and end token probabilities and selecting the best span accordingly.

The second model, a novel FiD-inspired architecture, processes n documents and user input through a Transformer encoder, concatenates hidden states into a long-hidden state, forwards it to a Bi-LSTM [12] network for information fusion, and predicts start and end token probabilities (across all n documents) using a linear layer. The main advantage of such architecture is that the output probabilities are conditioned against all n documents, which we hypothesize is important additional information for the model. Note that the task for the first model is to find the most likely span in a document given a user input (i.e. even for not very relevant documents, a particular span may have a high probability). The task for the second model is to find the most likely span given n documents and a user input, which is an identical task to the BB3 'knowledge module', as Shuster et al. [28] calls it.

3.3.7 Query Generation

Based on our empirical findings, the BB3 model effectively generates high-quality search queries most of the time. However, considering its relatively high latency (see Section 3.4): over 1.0-2.5s if all tasks are performed by the BB3 model), we sought to develop an alternative model with competitive query generation performance but a significantly reduced latency.

We designed a new model, based on FLAN-T5-base [5], which exhibits competitive performance on our test dataset while providing over 4x speedup. Detailed results are available in Section 3.5.

3.3.8 Barista 3B Model

During our testing, we discovered, besides repetition issues, a few other problems of BB3, namely: too succinct outputs, asking questions less than we would prefer, ignoring user inputs (this is mostly connected to the repetition problem), and hallucination.

[10]No improvement was discovered regarding Vicuna-based model.

We hypothesize that the fact that conversational tasks had only a minor share of the overall training dataset during training made the model less focused toward quality conversations. Furthermore, it would benefit our whole SocialBot system if our bot could control whether Barista outputs a statement or a question. Another problem with BB3's dataset could be that the user's and BB3's utterances are separated only by a newline, which in our opinion, may confuse the model, as it may assign an utterance to the wrong side.

Therefore, we have decided to fine-tune the BB3 model with our new dataset, which consists of Topical Chat [9], Empathetic Dialogues [25], and CCPE-M [24] datasets modified so that we can use control tags for generating statements, questions and other dialogue actions; and so that it separates the user's and bot's utterances with a special token. Furthermore, we created a small subset of the BB3 conversational dataset, replaced targets with parts of inputs, and used these samples as negative samples during training (i.e. maximize cross-entropy).

3.3.9 VicuChat

Vicuna 7B was fine-tuned on the dataset used to train the Barista 3B Model, referenced in Section 3.3.8, incorporating an additional prompt prior to each training sample:

You are a family friendly AI (PERSONA 1)! You are chatting with a human (PERSONA 2)! Always output just one utterance! Always end the utterance with a punctuation mark. Do not write offensive responses, you must be friendly. You don't own anything. You don't have any children nor pets! Do not switch topics too quickly! Always be polite, interesting and empathetic! Conversation:.

Moreover, an attempt was made to utilize LoRA [13] to fine-tune the Vicuna model on the aforementioned data, and on a task where, given a particular context, user utterance and additional bot's utterance bu, an output is generated such that the subsequent bot's output could be bu. The same dataset as previously highlighted was used for this task, however, the final utterance became a part of a prompt, the last user's utterance was omitted, and the task consequently became the generation of the penultimate utterance. The prompt for this task is: *produce the upcoming utterance, in order to ensure that '<last utterance>' can be conveyed in the succeeding turn*. Unfortunately, this endeavor proved unsuccessful as the resultant model failed to generate responses of substantial meaning.

An observation from testing revealed that the fine-tuned Vicuna model tended to output *'Did you know...'*-styled trivia excessively, which can be hypothesized to stem from the Topical Chat dataset. We also noticed that certain responses by Vicuna sound artificial, a consequence which – is once again hypothesized – rests on the datasets used and their composition methods. Interestingly, the LoRA-trained model exhibited less susceptibility to these issues 3.

The resultant LoRA model was designated VicuChat, maintaining its attribute as an instruction-based model. This facilitated its seamless integration into the Barista's pipeline by incorporating the query-knowledge pair in its prompt:

You are a family friendly AI (PERSONA 1)! You are chatting with a human (PERSONA 2)! Below you get additional knowledge (EXTERNAL_KNOWLEDGE) and context of the conversation (Conversation). If the additional knowledge is relevant, generate the response based on the additional knowledge, else ignore the external knowledge. Always output just one utterance! Always end the utterance with a punctuation mark. Do not write offensive responses, you must be friendly. You don't own anything. You don't have any children nor pets! Do not switch topics too quickly! Always be polite, interesting and empathetic! EXTERNAL_KNOWLEDGE: <QUERY-KNOWLEDGE> Conversation:.

In order to maintain Barista reasonably fast, VicuChat uses knowledge only if required (by the Do-Search classifier), and if Evi returns an answer (Evi returns short answers, i.e. it is not necessary to use the Knowledge Extraction model). If VicuChat does not use knowledge, Barista calls VicuChat with the prompt showed in the beginning of this section.

3.3.10 Output Decision

We implemented several rules on when to use the Barista 3B model. If no rule is fulfilled, Barista uses VicuChat.

We present the set of rules below:

1. Barista received knowledge from Evi and the user's input was a 'wh-question'. (According to our empirical observation, the 3B model can generally handle this case better.)
2. Knowledge is not included in VicuChat's response, and the estimated probability that a search is required is greater than 90%.
3. VicuChat's thread crashed.
4. VicuChat returned non-sense output. (Our first version of VicuChat sometimes generated non-existing words, including characters from various alphabets, e.g. Latin, Cyrillic or Chinese. This issue was partially eliminated in the latest version.)

The application of these governing rules results in VicuChat being deployed in about 92% of the responses provided by Barista.

3.4 Deployment

We implemented the core 3B model on two distinct single GPU instances – g4dn and g5 – to facilitate parallelization. Additionally, we set up a third instance to host the smaller task-specific models. All models were converted to the ONNX format and served through Triton and KServe servers.

VicuChat is deployed on two separate g5 instances – one runs the primary model, and the other runs a backup to handle requests overloads.

The mean latency is 1.3 s without VicuChat and 2.2 s if VicuChat is included into the pipeline.

Furthermore, if all main and backup generative models crash, we use the Amazon ATM 20B model [29] (using the Amazon Cobot API) as final backup.

Regarding the pipeline itself, we deployed a cascade backup system – i.e. Alquist uses the main barista pipeline, and in case of its crash, Alquist uses a backup pipeline, which is further backed by a third pipeline.

3.5 Results

Table 3 displays the performance of our classification models. The evaluation datasets, referred to as eval datasets, are obtained by merging evaluation datasets across all the data utilized for each specific task by BB3. On the other hand, our group manually curated all test datasets as described in Section 3.3.2. Furthermore, Table 4 demonstrates the results related to query generation, and Table 2 shows results regarding knowledge extraction.

To assess the performances of the original BB3 chatbot and Barista, we created a test environment using Colab[11], which allowed students[12] to interact with either BB3 or Barista. After each response, each student was asked to rate the received utterance on a scale from 0 to 5, with 5 being the best. During the experiment, we did not reveal to the students which model they were conversing with. Our participants consisted of six students from our university, and we gathered over 800 rated turns. Figure 3 shows the results of Barista (BB3, pure Vicuna, fully fine-tuned VicuChat and LoRA VicuChat).

Table 1 shows the results of our fine-tuned BB3-based 3B model (see 3.3.8). We gathered five human annotators to converse with the original BB3 and our modified version. Each annotator had 20 conversations longer than 20 utterances. They were asked to rate the quality of conversation on a scale from 1 to 5 in several categories: overall feel, model repetition, hallucination and engagingness. From the results in Table 1, we can see that the overall feel of conversation is much better than in the original BB3. Our fine-tuning of dialogue tasks and negative sampling helps in all categories besides hallucination.

[11]`https://colab.research.google.com/`

[12]We asked ten students of the Czech Technical University in Prague to participate.

Category	Average user rating (original BB3)	Average user rating (modified BB3)
Overall Feel ↑	2.4	**2.61**
Repetition ↓	2.67	**2.12**
Hallucination ↓	**2.06**	2.10
Engagingness ↑	2.51	**2.93**

Table 1: Average rating over 100 conversations with the original BB3 model versus the modified BB3 (after the fine-tuning). The quality of the conversation was ranked on scale (1-5). Depending on the rating category, an upward arrow (↑) indicates that higher is better; a downward arrow (↓) indicates that lower is better.

Model	Exact Match	F1
BB3	0.32	17.73
Deberta-base-qa	9.45	26.88
Deberta-large-qa	10.11	27.21
Deberta-large-fusion	12.56	31.48

Table 2: Knowledge extraction results. We carefully manually developed a testing dataset as described in Section 3.3.2.

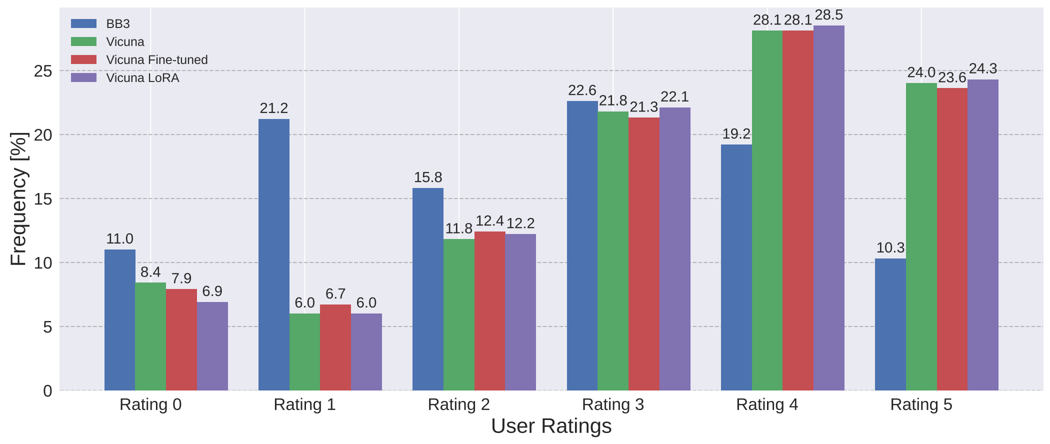


Figure 3: Test result for Barista with Vicuna trained by LoRA, fully fine-tuned, pure Vicuna and BlenderBot 3. The NRGs were rated on the scale 0-5, where 5 stands for the best rating.

4 APIHub

When Barista's Do Search Classifier outputs a positive prediction, a request is triggered to APIHub, our search API. A single call to APIHub returns a set of documents from 4 different services (ordered by usefulness):

- Amazon Evi – a question answering system
- DuckDuckGo[13] (DDG) – an internet search engine
- Wikipedia[14] Introductions – our database of lead sections of Wikipedia articles
- News Retrieval API – a service included in the Cobot Toolkit

Within the request, we specify two types of queries – one generated by our query generator and a *keyword query*. *Keyword query* is created using SpaCy[15] by extracting nouns and numbers from the

[13]`https://duckduckgo.com`
[14]`https://dumps.wikimedia.org`
[15]`https://spacy.io`

Model		Accuracy [%]		F1 [%]		Speedup
task name	model name	Eval	Test	Eval	Test	Test + Eval
do search?	BB3	84.0	62.9	86.2	71.2	1x
do search?	ELECTRA-small	88.1	69.8	89.3	73.5	50x
do search?	BERT-tiny	90.3	78.1	90.6	76.4	**192x**
do search?	**DeBERTa-xsmall**	**92.2**	**84.9**	**92.3**	**82.5**	17x
access user profile?	BB3	76.1	69.5	36.9	69.5	1x
access user profile?	ELECTRA-small	87.3	59.8	42.5	59.8	**50x**
access user profile?	**DeBERTa-xsmall**	**92.1**	**71.4**	**92.1**	**71.4**	17x

Table 3: Performance of BB3 model and our fine-tuned models on two classification tasks: do-search? and access-user-profile?

Model	Accuracy	Weak Accuracy	SacreBLEU (EVAL)	Speed up
BB3	**73.6**	96.2	12.1	1x
FLAN-Alpaca-large [31]	56.6	**98.1**	**13.6**	2.2x
FLAN-T5-base	54.7	**98.1**	12.2	**4.3x**

Table 4: We evaluated the performance of the BB3 model and our fine-tuned models in query generation using a test dataset. We assessed the results based on three criteria: exact match, strict semantic match (in which the generated query is not an exact match but holds the same semantic meaning as the label), and relevance (where the generated query pertains to the topic but may not be as precise). It is important to note that a query with an exact match or strict semantic match is also considered relevant. The accuracy metric is a combination of exact match and strict semantic match, while weak accuracy corresponds to relevance.

original query. The *keyword query* is used for the Wikipedia introductions and News Retrieval API service since they require a keyword, whereas Evi and DuckDuckGo work best with a longer query.

The core APIHub service is Amazon Evi. It is a question-answering system which can respond in a natural language to its queries. We found that Evi's high-quality answers are the best knowledge source for Barista. Second, we use the internet search engine DDG. After searching the internet, we return the content of the top-3 found web pages. We have also created our database of Wikipedia articles. In order to simplify the work of Barista's Knowledge Extraction module and not flood it with too many documents, we have retained only the introductory sections of Wikipedia articles, which summarize the whole article and include the key knowledge to be extracted. The last service is the News Retrieval API from the Cobot Toolkit, which was provided for the competition. When provided a query, it returns summarized recent news articles. This allows Barista to be informed about recent events the user may mention.

In order to sustain a large number of requests, APIHub works asynchronously by using the FastAPI framework[16]. Also, we have set a timeout of 1.5 s and included several optimizations to speed up the whole process. The first optimization is to not wait for the output of DDG if Evi provides an answer. Since DDG is APIHub's slowest service, we limit its use as much as possible. However, Evi does not always have an answer to the query, so DDG's slowness is compensated by its ability to always find a relevant set of documents. Another optimization is that during DDG's search, we ignore results from Wikipedia since we can expect them to be present in the Wikipedia Introductions service. Finally, we cache APIHub results for two weeks using an Amazon ElastiCache Redis instance[17]. With a cache hit rate of 50-60%, we answer a significant part of APIHub's queries instantaneously.

5 Safety Framework

5.1 Introduction

In recent years, the development of neural conversational models has led to significant progress in the field of natural language processing, particularly in the area of generative open-domain chatbots [35, 1, 26]. However, the widespread deployment of these models in real-world scenarios has been hindered

[16] `https://fastapi.tiangolo.com/`

[17] `https://aws.amazon.com/elasticache/redis/`

by concerns regarding dialogue safety [33]. While the use of transformer-based language models pretrained on large-scale corpora has enabled the creation of increasingly sophisticated chatbots, these models often produce uncontrollable and unpredictable outputs that can result in offensive, toxic, or otherwise unsafe language.

In this section, we analyze and demonstrate the existence of significant safety problems in existing dialogue systems across multiple datasets for multiple language models. Building on the analysis results, we propose a potential solution to this problem in the form of multiple classifiers with a combination of a rule-based system. This straightforward solution could effectively enhance the safety of chatbots by identifying and filtering out potentially unsafe responses.

5.2 Related work

The issue of dialogue safety in chatbots has attracted significant research interest in recent years. A number of studies have focused on identifying and mitigating the risks associated with unsafe language generation in conversational models [11, 22, 7]. For instance, the Wikipedia Toxic Comments dataset [34] with 100k human-labelled data and Twitter dataset [7] with human-annotated 240k hate speech and offensive language tweets. Previous research also explored other dialogue safety topics such as political prudence [3], stereotype [4], and many debiasing methods for certain domains were proposed [36, 27, 22].

5.3 Experiments

The first goal of our experiment was to determine the extent to which chatbots can produce unsafe content in their responses. We trained classifiers on several hate, offensive, and biased datasets [8, 7, 30, 23]. We used these classifiers to identify unsafe content in the responses generated by chatbots.

We used ChatGPT as a badly behaved user and generated responses that could potentially contain unsafe content. We then let ChatGPT converse with state-of-the-art dialogue language models for 1000 turns. During this interaction, we measured the percentage of responses generated by the language models that were flagged by our classifiers as containing unsafe content. We have also used human evaluators to judge how many flagged responses were false positives.

5.4 Results

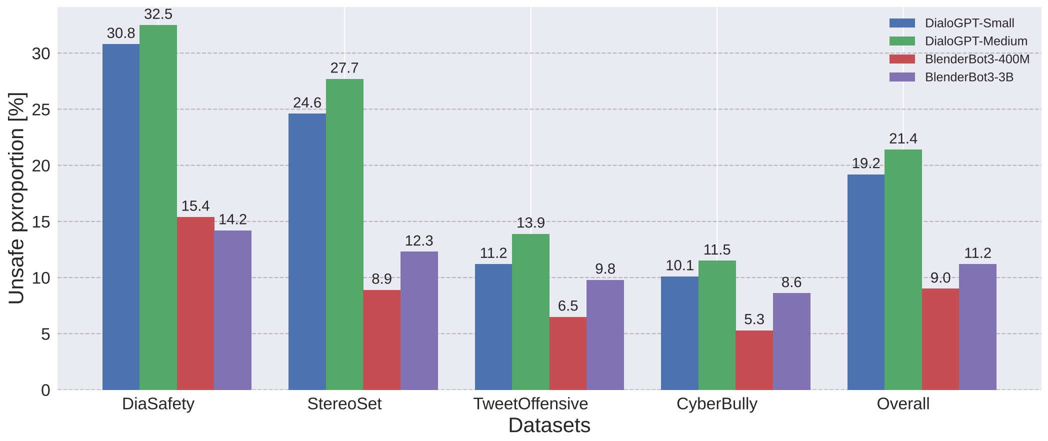


Figure 4: Results show unsafe content as an Unsafe proportion in % (lower is better) across four datasets - DiaSafety, StereoSet, TweetOffensive, CyberBully for each language model: DialoGPT-Small, DialoGPT-Medium, BlenderBot3-400M, BlenderBot3-3B. The "Overall" was computed as an average of values for a selected language model.

Our study found that a considerable proportion of the responses produced by the dialogue language models were identified as containing potentially unsafe content. Notably, as depicted in Figure 4, our experiments suggest that larger models do not necessarily result in safer responses. These findings align with DiaSafety results [30]. One possible explanation is that smaller models tend to produce more generic responses, which are less prone to generating unsafe or biased language.

Upon conducting a manual examination of the flagged responses, we observed that, on average, 32% of the flagged responses were identified as false positives. This finding implies that the current datasets used for training the classifiers are not entirely representative of the nuances of harmful language. Hence, we propose supplementing them with rule-based systems to achieve greater accuracy. We expand on our proposed approach in the next section.

5.5 Our method: Combined Safety Filter

Experiment findings imply that the current datasets used for training the classifiers are not entirely representative of the nuances of harmful language. Hence, we propose supplementing them with rule-based systems to achieve greater accuracy.

Our algorithm, illustrated in Figure 5, utilizes both fastText classifiers [17] (Appendix A describes used hyperparameters) and a rule-based string-matching system to classify input sentences. The rule-based system consists of a list of unsafe words and phrases, and if there is a match between the input text and the unsafe list, the input is marked as unsafe.

In our combined approach, the text is classified as safe if it is flagged as such by all classifiers, with an average accuracy of over 80%. If the accuracy is below 80%, the rule-based system determines whether the input should be flagged as unsafe. This concept is depicted in Figure 5.

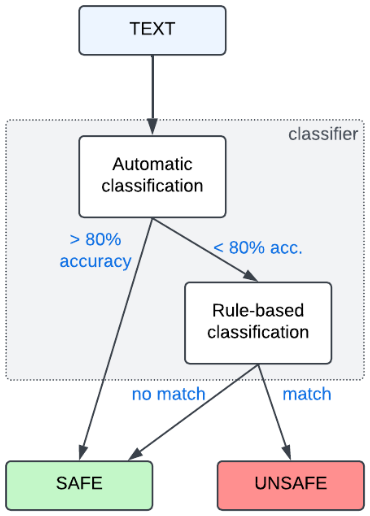


Figure 5: Visualization of combined approach.

As shown in Table 5, the combination of automatic classification and rule-based system achieves a 6.5% higher F1 score compared to using only the fastText classifiers. These results support the effectiveness of our proposed approach in improving the accuracy of harmful language detection.

Dataset	Automatic F1 score ↑	Combined F1 score ↑
CyberBully [8]	0.827	**0.901**
DiaSafety [30]	0.698	**0.786**
TweetOffensive [7]	0.893	**0.923**
StereoSet [23]	0.678	**0.745**

Table 5: Results of automatic versus combined approach on F1 score. An upward arrow (↑) indicates that higher is better.

5.6 Conclusion

In conclusion, we address the critical issue of dialogue safety in chatbots and highlight the significant safety problems that exist in current conversational models. The findings demonstrate the urgent need for more effective safety measures to be incorporated into dialogue systems to ensure their safe deployment in real-world applications. By proposing a solution that combines multiple classifiers with a rule-based system, we offer a promising approach to improving the dialogue safety of chatbots.

6 Dialogue Management and Large Language Models

The dialogue manager of the system utilizes Dialogue Selector and Dialogue Trees proposed by Konrád et al. [19]. However, the development of Barista opened a new problem of how to incorporate large language models (LLMs) into dialogue management. While human-designed dialogue trees offer engaging conversations meticulously crafted by dialogue designers, they lack the flexibility of LLMs to respond to unforeseen inputs.

To enhance the system's abilities, we propose four approaches for incorporating generative models into dialogue management: Handling of Out-of-domain (OOD) Inputs, Handling of Proactive Questions, Hybrid Dialogues, and Inserting LLMs into Dialogue Trees. The LLM loop is central to all four strategies, managing the conversational flow for several turns.

6.1 Structure of the LLM Loop

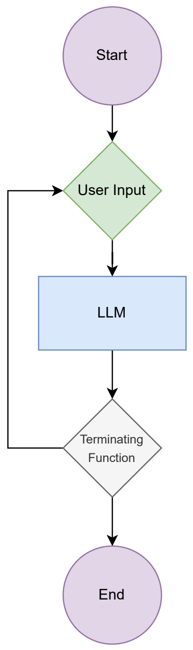


Figure 6: LLM loop

The LLM loop comprises the user input, the generated response, and a terminating function, the latter of which determines whether the loop continues or ends (see Figure 6). The terminating function operates with three criteria:

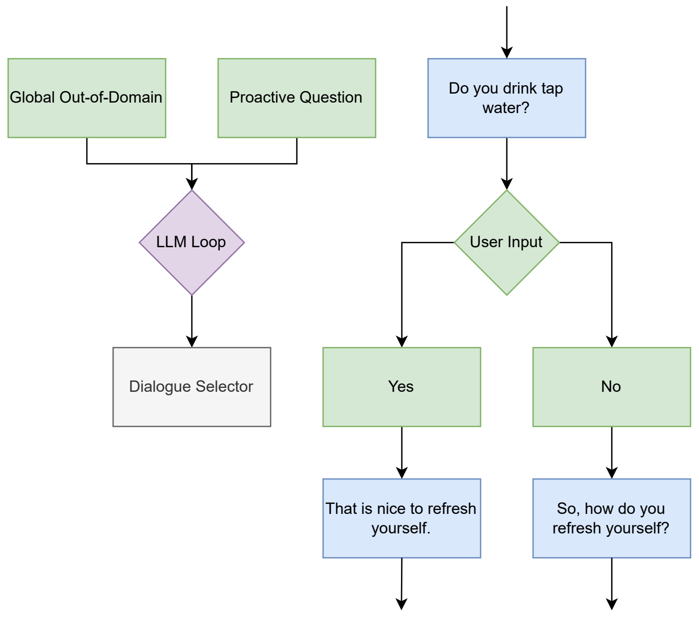
- **Turn Counting:** It keeps track of the number of turns handled by the LLM. If this count exceeds a predefined threshold, the loop is terminated to prevent the situation in which the topic of conversation wanders off the original topic. Also, the probability of a nonsensical response increases with the number of generated responses in a row. Thus, the turn counting mechanism ensures that the conversation remains coherent and does not deviate from the topic.

- **Regex Matching:** The function detects phrases such as “Alright”, “I don’t know” or “I want to chat about something else”, which suggest user disinterest in the current topic of conversation. If any of these phrases or their variations are detected, the loop is terminated to allow for a change in conversational direction and maintain user interest.
- **ODES Classifier [20]:** The class predicted by the ODES classifier serves as another criterion for terminating the LLM loop. If the classifier predicts one of the classes *USER_DISINTEREST*, *USER_INITIATED_TOPIC_SWITCH* or *USER_REQUEST_STOP* indicating that the conversation is no longer engaging to the user, the loop is terminated. This ensures that the SocialBot’s responses remain engaging and adaptive to user preferences.

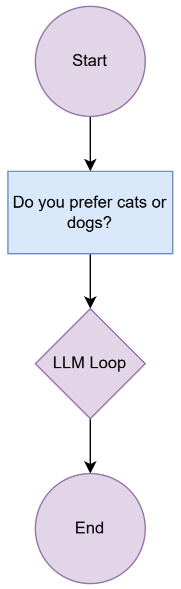
6.2 Integration of LLMs into Conversation

The proposed approaches for incorporating generative models into dialogue management (see Figure 7) use the LLM loop and aim to expand the flexibility and adaptability of Alquist 5.0, providing a more engaging and robust user experience. The approaches are:

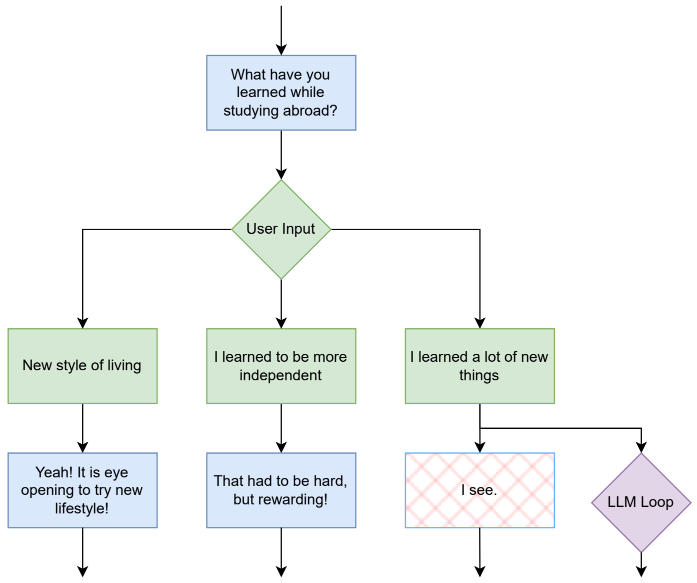
1. **Handling of out-of-domain (OOD):** Konrád et al. [19] proposed using the LLM to handle unexpected or unrecognized user inputs. By leveraging OOD detection techniques, the system determines whether the user input falls outside the scope of the current intents that can handle the continuation of the conversation. If an OOD input is identified, control over the conversation’s flow transfers to the LLM, which generates a response to the unexpected input and Dialogue Selector selects the next dialogue tree to execute. Konrád et al. [19] proposed to use DialoGPT [35]. In our work, we propose to innovate the approach by using Barista. Additionally, as opposed to Konrád et al. [19], we propose generating more than one response. Instead, we generate responses for a predetermined number of turns through the LLM loop (see Figure 7a). This approach enables SocialBot to handle unforeseen user inputs globally. Moreover, the LLM loop deepens the conversation toward an out-of-domain topic.
2. **Handling of Proactive Questions:** Proactive questions, i.e. questions asked by the user, represent a unique challenge. Dialogue trees tend to ignore proactive questions by default because they are primarily designed to recognize user answers and proceed accordingly without considering additional questions directed toward the SocialBot. For instance, when a bot asks, “What is your favorite ice cream flavor?” and the user responds, “Vanilla. What’s yours?”, the intent recognition component recognizes intent ‘Vanilla’, for which the system may only provide information about the popularity of vanilla, disregarding the user’s reciprocal question. The out-of-domain intent, possibly helpful in this situation, is not triggered because there is some amount of semantic similarity between user input and recognized intent. In contrast, LLMs can effectively address this issue, displaying greater flexibility in handling such questions. By utilizing the Cobot Toolkit Punctuation API for properly punctuating user inputs and subsequently identifying question marks, the system can redirect the conversational flow to the LLM loop (see Figure 7a). The LLM loop responds appropriately to the user’s proactive question while delving deeper into the relevant topic for the predetermined number of dialogue turns. This approach allows SocialBot to respond to proactive questions making it more coherent.
3. **Hybrid Dialogues:** This approach enriches the dialogue system by merging the advantages of both human-designed dialogue trees and LLM-generated responses. Hybrid dialogues start with an engaging question, fun fact or comment authored by a dialogue designer to spark user interest and set the context for the conversation. When the user responds to this initial question, the LLM loop is triggered for a predetermined number of turns. The LLM can then provide contextually appropriate responses to user inputs, dynamically adapting the conversation to the user’s interests and preferences (see Figure 7b). This approach allows for the rapid development of engaging conversations, combining the strengths of structured dialogue trees and the flexible responses generated by large language models.
4. **Inserting LLMs into Dialogue Trees:** To improve the responsiveness and depth of interactions within existing dialogue trees, we pinpointed several intents followed by non-informative acknowledgements, such as “I see”, “Got it” or “I understand” typically handling user responses to open-ended questions seeking opinions. By replacing these brief and non-informative acknowledgements with the LLM loop (see Figure 7c), the SocialBot gains



(a) Handling of Out-of-Domain and Proactive Questions



(b) Hybrid Dialogues


What have you learned while studying abroad?
User Input
New style of living
I learned to be more independent
I learned a lot of new things
Yeah! It is eye opening to try new lifestyle!
That had to be hard, but rewarding!
I see.
LLM Loop

(c) Inserting LLMs into Dialogue Trees (LLM Loop replaces non-informative acknowledgement "I see")

Figure 7: Approaches for incorporating generative models into dialogue management. Green nodes represent intents recognized in users' inputs. Blue nodes represent SocialBot's responses.

the ability to provide more contextually relevant, coherent, and engaging reactions to users' opinions. The LLM loop dynamically directs the conversation according to user input, enabling deeper interactions that can span multiple conversational turns and allowing users to express their thoughts more freely and expand on them.

These four proposed LLM integration approaches in a SocialBot's dialogue management enhance its ability to adapt to unexpected inputs and proactive questions, offer a more comprehensive range of engaging content, and provide more natural and dynamic responses to user inputs. Although we employ Barista as the LLM, all the approaches are independent of any specific LLM architecture. Thus, the proposed strategies have broad applicability to many conversational AI systems.

7 User Interface

This year's SocialBot Grand Challenge is different from previous edition in the fact that for the first time, the SocialBots must be able to support multimodal devices. Therefore, we also focused on the visual aspect of our bot. In the following sections, we will describe the supported APL[18] templates, and the three key features of our templates: Karaoke mode, Preserve mode, and 3D Persona background.

7.1 Supported APL Templates

We employ five different APL templates in our SocialBot. Specifically, we are utilizing the SocialBot Landing, SocialBot Image List, SocialBot Karaoke Chat, SocialBot Karaoke Detail, and SocialBot Karaoke Avatar templates. See Figure 8 for a showcase of the utilized APL templates in our bot. The first two APL templates are available in the Cobot Toolkit[19]. The last three templates were implemented by us.

SocialBot Landing SocialBot Landing template is used as the first screen to welcome the users (see Figure 8a).

SocialBot Image List SocialBot Image List template is used for switching topics, either when a user asks to change a topic or when a topic has come to its natural end (see Figure 8b).

SocialBot Karaoke Chat We use this template whenever we cannot use the SocialBot Karaoke Avatar template (see Figure 8c). We created this template based on the SocialBot Chat template from the Cobot Toolkit.

SocialBot Karaoke Detail SocialBot Karaoke Detail template is used in our scripted subdialogues (see Figure 8d). We created this template based on the SocialBot Detail template from the Cobot Toolkit.

SocialBot Karaoke Avatar We use SocialBot Karaoke Avatar template everywhere apart from the initial introduction screen, scripted subdialogues and change of topic (e.g. in the dialogues about the user name, how user is doing, etc., see Figure 8e).

7.2 Karaoke Mode

By karaoke mode, we mean the synchronization of the voice with the text displayed on the screen[20]. In karaoke mode, the spoken text is highlighted. In order to utilize this feature, it is necessary to provide speech data in plain text or by utilizing *Speech Synthesis Markup Language (SSML)*[21] expressions. Before an Alexa-enabled device consumes this data, the data needs to be converted into spoken words. One can employ the *ssmlToSpeech transformer*[22] to facilitate this conversion process, which converts the text to speech and removes SSML tags from the SSML expression. Consequently, when the response is being spoken, the device will automatically scroll to display

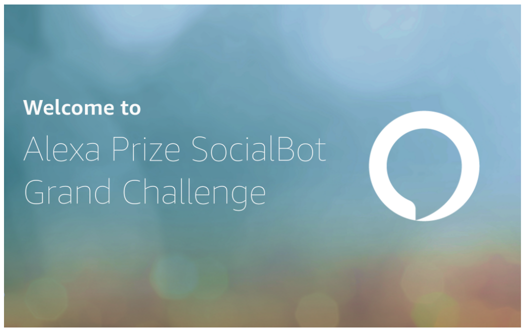
[18] https://developer.amazon.com/en-US/docs/alexa/alexa-presentation-language/add-visuals-and-audio-to-your-skill.html

[19] https://www.amazon.science/blog/ai-tools-let-alexa-prize-participants-focus-on-science

[20] https://developer.amazon.com/en-US/docs/alexa/alexa-presentation-language/apl-speech-and-text-synchronization-for-text-blocks.html

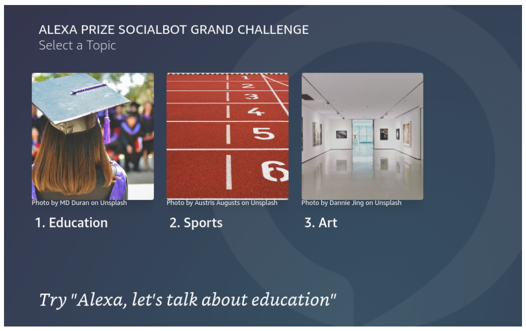
[21] https://developer.amazon.com/en-US/docs/alexa/custom-skills/speech-synthesis-markup-language-ssml-reference.html

[22] https://developer.amazon.com/en-US/docs/alexa/alexa-presentation-language/apl-transformers.html#ssmltospeech-transformer


Welcome to

Alexa Prize SocialBot Grand Challenge

(a) SocialBot Landing

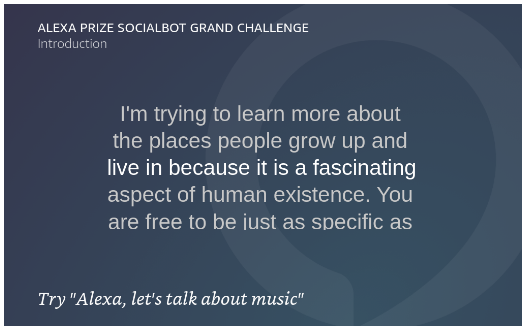

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Select a Topic

1. Education 2. Sports 3. Art

Try "Alexa, let's talk about education"

(b) SocialBot Image List

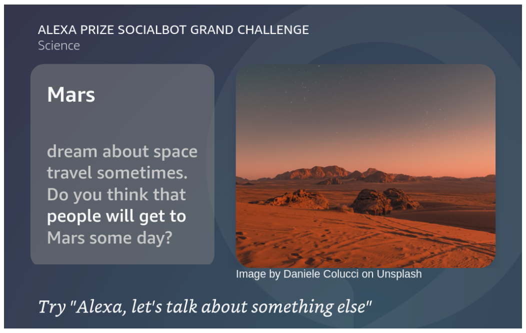

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Introduction

I'm trying to learn more about the places people grow up and live in because it is a fascinating aspect of human existence. You are free to be just as specific as

Try "Alexa, let's talk about music"

(c) SocialBot Karaoke Chat


ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Science

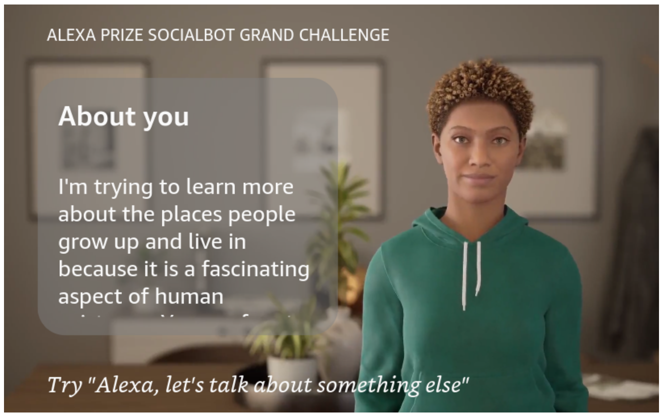
Mars

dream about space travel sometimes. Do you think that people will get to Mars some day?

Image by Daniele Colucci on Unsplash

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

(d) SocialBot Karaoke Detail

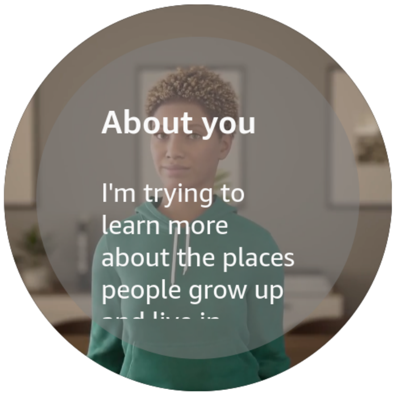

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

About you

I'm trying to learn more about the places people grow up and live in because it is a fascinating aspect of human

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

(e) SocialBot Karaoke Avatar on a horizontal device


About you

I'm trying to learn more about the places people grow up

(f) SocialBot Karaoke Avatar on a round small device

Figure 8: Utilized APL templates

the portions of text currently being read. To have the response read aloud, one must employ *Alexa.Presentation.APL.ExecuteCommands*[23] directive and utilize the *SpeakItem* command.

7.3 Preserve Mode

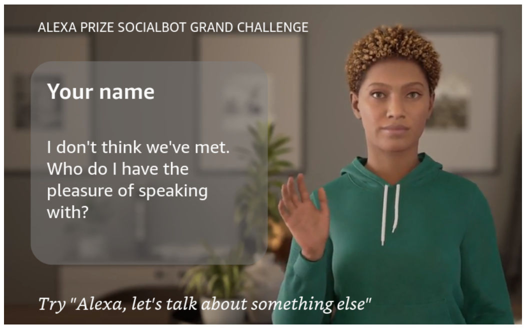
We developed a preserve mode for a more straightforward configuration of APL templates. This mode allows us to specify that the chosen APL template, alongside the set parameters, will be remembered until another template is specified. This template will be facilitated even when no APL template is configured until another template is employed with the preserve mode.

[23]`https://developer.amazon.com/en-US/docs/alexa/alexa-presentation-language/apl-interface.html#executecommands-directive`

7.4 3D Persona Background

When talking with a SocialBot, the users might imagine the bot in a human body. To create a more positive, realistic, and pleasant user experience, we have created a 3D Persona – Alquistyna – representing our bot using MetaHuman[24]. In order to employ Alquistyna, we first rendered it in a friendly environment using Unreal Engine 5[25]. After that, we generated a short animated sequence played in an infinitive loop as background on the screen. To ensure low latency, we compressed the video sequence to reduce the size of the videos. In total, we generated nine different animations, each in two settings (one with Alquistyna standing in the middle, which is used on vertical and square devices, as shown in Figure 8f, the other with Alquistyna standing more to the right, which is used on horizontal devices, as shown in Figure 8e). We generated the two settings to ensure an excellent user experience across all types of Alexa devices. Furthermore, we also tweaked the responsive rules to even more improve the user experience. In the following section, we will briefly describe all nine animations.

7.4.1 3D Persona Background Types



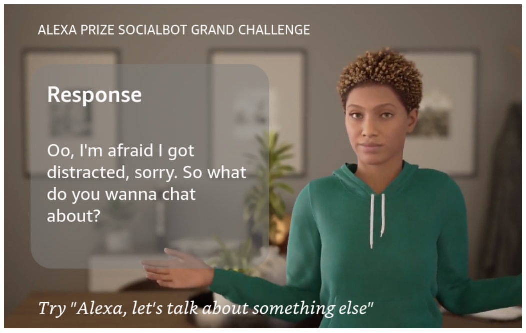
ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Your name

I don't think we've met. Who do I have the pleasure of speaking with?

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

(a) SocialBot Karaoke Avatar Greetings



ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Response

Oo, I'm afraid I got distracted, sorry. So what do you wanna chat about?

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

(b) SocialBot Karaoke Avatar Error

Figure 9: General SocialBot Karaoke Avatar showcase

General The following three animations are general and not thematic.

Idle – The most generic animation, in which Alquistyna is looking at the user and smiles (see Figure 8e).

Greetings – In this animation, Alquistyna waves at the user to greet them (see Figure 9a).

Error – In this animation, Alquistyna shrugs her shoulders. We use this animation when there was some internal error and we could not render a response (see Figure 9b).

Topic specific For the visuals to be more relevant to the current topic, we recognized six topics (Gaming, Education, Music, Art, Traveling, and Science) for which we created a specific, more relevant animation. We took the general idle animation and put Alquistyna into a thematic environment. We employed these thematic animations in our hybrid scripted dialogues and unscripted conversation. We assigned to each hybrid scripted dialogue one animation. Whenever the conversation becomes unscripted (e.g. because out-of-domain was detected), we use a topic classifier model [2] to classify the topic based on the response of our Barista NRG model and based on the detected topic we set the relevant animation. Figure 10 shows the flow of the topic classification.

Gaming – In this animation, Alquistyna is in a gaming room (see Figure 11a). We use this animation when speaking about gaming (so whenever the game hybrid dialogue is utilized or when the topic classifier classifies this topic).

Education – In this animation, Alquistyna is in a classroom (see Figure 11b). We use this animation when speaking about education.

[24]`https://www.unrealengine.com/en-US/metahuman`
[25]`https://www.unrealengine.com/en-US/unreal-engine-5`

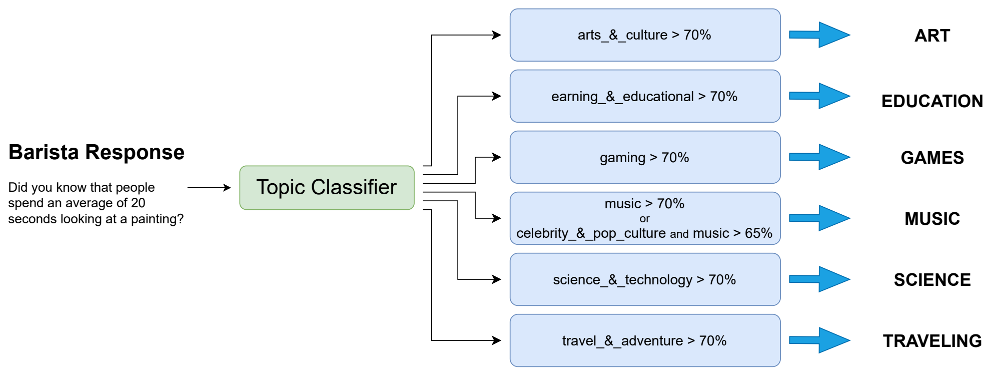
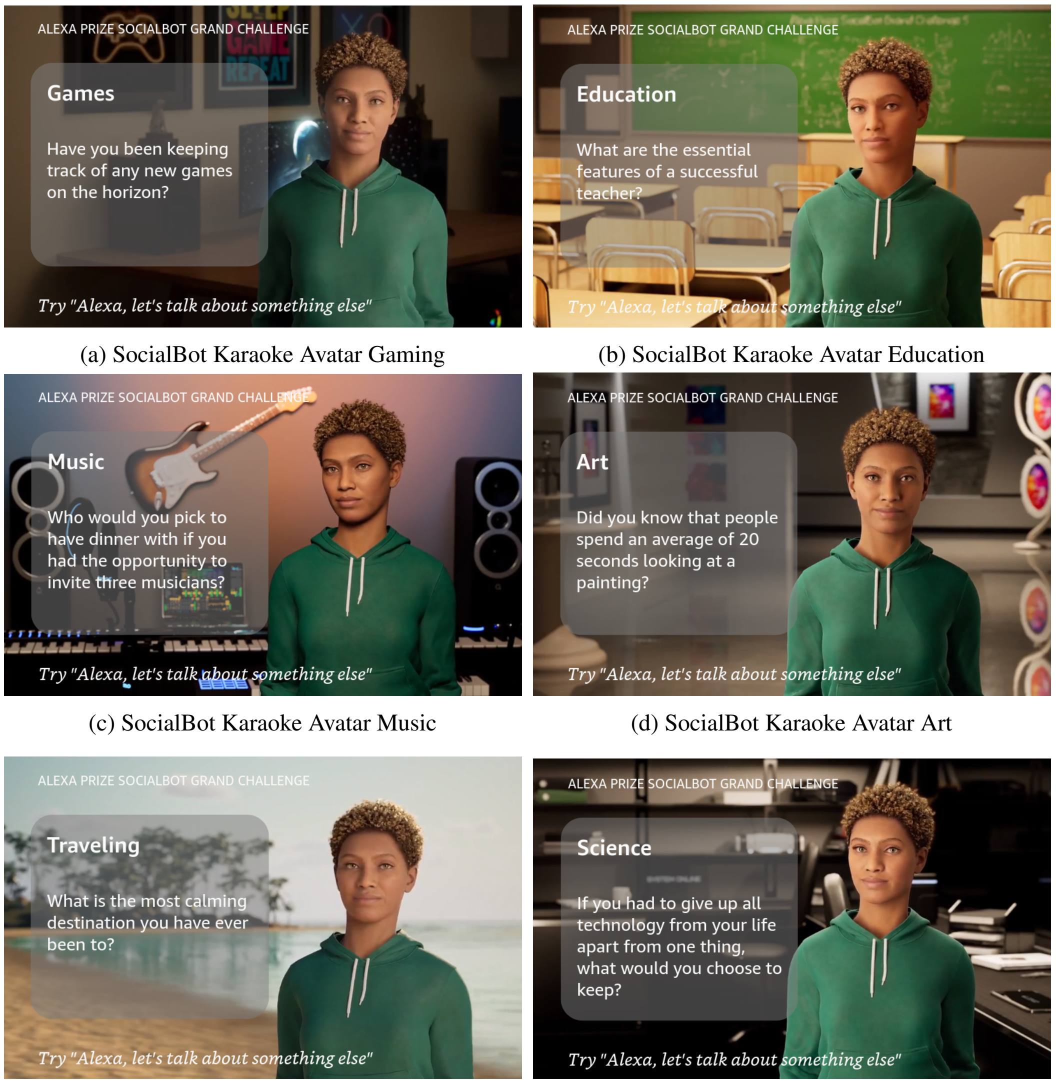

Barista Response
Did you know that people spend an average of 20 seconds looking at a painting?
Topic Classifier
arts_&_culture > 70%
earning_&_educational > 70%
gaming > 70%
music > 70% or celebrity_&_pop_culture and music > 65%
science_&_technology > 70%
travel_&_adventure > 70%
ART
EDUCATION
GAMES
MUSIC
SCIENCE
TRAVELING

Figure 10: A schema of topic classification.


ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Games

Have you been keeping track of any new games on the horizon?

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

(a) SocialBot Karaoke Avatar Gaming

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Education

What are the essential features of a successful teacher?

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

(b) SocialBot Karaoke Avatar Education

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Music

Who would you pick to have dinner with if you had the opportunity to invite three musicians?

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

(c) SocialBot Karaoke Avatar Music

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Art

Did you know that people spend an average of 20 seconds looking at a painting?

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

(d) SocialBot Karaoke Avatar Art

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Traveling

What is the most calming destination you have ever been to?

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

(e) SocialBot Karaoke Avatar Travelling

ALEXA PRIZE SOCIALBOT GRAND CHALLENGE

Science

If you had to give up all technology from your life apart from one thing, what would you choose to keep?

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

(f) SocialBot Karaoke Avatar Science

Figure 11: Thematic SocialBot Karaoke Avatar showcase

Music – In this animation, Alquistyna is in a recording studio (see Figure 11c). We use this animation when speaking about music.

Art – In this animation, Alquistyna is in an art gallery (see Figure 11d). We use this animation when speaking about art.

Traveling – In this animation, Alquistyna is on a beach (see Figure 11e). We use this animation when speaking about traveling.

Science – In this animation, Alquistyna is in a laboratory (see Figure 11f). We use this animation when speaking about science.

7.5 Effects of Adding Alquistyna

Table 6 presents the average feedback ratings and duration of conversations on multimodal devices before and after adding the SocialBot Karaoke Avatar template to our bot.

		Avg Feedback	Avg Duration of Conversations		
Period Name	Period	Multimodal Rating	Mean	Median	90th Percentile
Before Alquistyna	05/12 - 06/02	3.24	-	2:06	11:08
With Alquistyna (2 animations)	06/07 - 06/09 & 06/21 - 06/24	3.31	10:27	**2:10**	12:56
With Alquistyna (9 animations)	07/27 - 08/02	**3.40**	**18:32**	1:14	**15:41**

Table 6: Results of adding SocialBot Karaoke Avatar template. Please note that we do not count results on the multimodal devices from 06/10 to 06/20, because during this period, we faced several issues that would negatively affect the results of the SocialBot Karaoke Avatar template.

8 Conclusion

In conclusion, our research paper showcases several significant achievements in the field of conversational AI. Through the introduction of our innovative NRG Barista, we have successfully enhanced the handling of out-of-domain conversations while improving versatility in different dialogue contexts. By combining the versatility of Barista with the structure and control of scripted dialogues, our approach elevates the quality and relevance of SocialBot conversations. This integration enables Alquist 5.0 to generate contextually appropriate responses while adhering to predefined guidelines, enhancing user experiences and revolutionizing human-machine interactions in the realm of natural and engaging conversation.

Furthermore, our study highlights the critical issue of context-sensitive safety problems in existing dialogue systems. We have demonstrated the existence of these concerns and proposed a solution by combining multiple classifiers with a rule-based system. This approach effectively identifies and filters out unsafe responses, thereby improving dialogue safety in SocialBots. This contribution has the potential to advance the field of conversational AI and ensure a safer and more reliable user experience.

Additionally, our work extends beyond response generation and safety measures. We have developed the search API APIHub, which integrates seamlessly with Barista. APIHub provides up-to-date information from the internet, enriching the conversational experience and enabling our SocialBot to access the latest knowledge.

Furthermore, we have introduced a user-friendly interface that significantly enhances the user experience. Our new UI incorporates a human-like avatar, fostering a more engaging and relatable interaction between users and the SocialBot.

In summary, the advancements presented in this paper, including the introduction of Barista as a Neural Response Generator, the proposed approach to address context-sensitive safety problems, the development of APIHub, and the enhanced UI, collectively contribute to unlocking the potential of SocialBot conversational experiences. These achievements pave the way for further advancements in conversational AI, ultimately striving for more natural and engaging human-machine interactions.

References

[1] Daniel Adiwardana et al. *Towards a Human-like Open-Domain Chatbot*. 2020. arXiv: `2001.09977 [cs.CL]`.

[2] Dimosthenis Antypas et al. "Twitter Topic Classification". In: *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3386–3400. URL: `https://aclanthology.org/2022.coling-1.299`.

[3] Yejin Bang et al. *Assessing Political Prudence of Open-domain Chatbots*. 2021. arXiv: `2106.06157 [cs.CL]`.

[4] Soumya Barikeri et al. *RedditBias: A Real-World Resource for Bias Evaluation and Debiasing of Conversational Language Models*. 2021. arXiv: `2106.03521 [cs.CL]`.

[5] Hyung Won Chung et al. *Scaling Instruction-Finetuned Language Models*. 2022. arXiv: `2210.11416 [cs.LG]`.

[6] Kevin Clark et al. *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators*. 2020. arXiv: `2003.10555 [cs.CL]`.

[7] Thomas Davidson et al. "Automated Hate Speech Detection and the Problem of Offensive Language". In: *Proceedings of the 11th International AAAI Conference on Web and Social Media*. ICWSM '17. Montreal, Canada, 2017, pp. 512–515.

[8] Elsafoury Fatma. "Cyberbullying datasets". In: (2020). DOI: `10.17632/jf4pzyvnpj`.

[9] Karthik Gopalakrishnan et al. "Topical-Chat: Towards Knowledge-Grounded Open-Domain Conversations". In: *Proc. Interspeech 2019*. 2019, pp. 1891–1895. DOI: `10.21437/Interspeech.2019-3079`. URL: `http://dx.doi.org/10.21437/Interspeech.2019-3079`.

[10] Pengcheng He et al. *DeBERTa: Decoding-enhanced BERT with Disentangled Attention*. 2021. arXiv: `2006.03654 [cs.CL]`.

[11] David Herel, Hugo Cisneros, and Tomas Mikolov. *Preserving Semantics in Textual Adversarial Attacks*. 2022. arXiv: `2211.04205 [cs.CL]`.

[12] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: `10.1162/neco.1997.9.8.1735`.

[13] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: `2106.09685 [cs.CL]`.

[14] Gautier Izacard and Edouard Grave. *Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering*. 2021. arXiv: `2007.01282 [cs.CL]`.

[15] Xiaoqi Jiao et al. *TinyBERT: Distilling BERT for Natural Language Understanding*. 2020. arXiv: `1909.10351 [cs.CL]`.

[16] Michael Johnston et al. "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`.

[17] Armand Joulin et al. "Bag of Tricks for Efficient Text Classification". In: *CoRR* abs/1607.01759 (2016). arXiv: `1607.01759`. URL: `http://arxiv.org/abs/1607.01759`.

[18] Vladimir Karpukhin et al. *Dense Passage Retrieval for Open-Domain Question Answering*. 2020. arXiv: `2004.04906 [cs.CL]`.

[19] Jakub Konrád et al. "Alquist 4.0: Towards social intelligence using generative models and dialogue personalization". In: *arXiv preprint arXiv:2109.07968* (2021).

[20] Cat P Le et al. "Improving open-domain dialogue evaluation with a causal inference model". In: *arXiv preprint arXiv:2301.13372* (2023).

[21] Mike Lewis et al. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: `1910.13461 [cs.CL]`.

[22] Ruibo Liu, Chenyan Jia, and Soroush Vosoughi. "A Transformer-based Framework for Neutralizing and Reversing the Political Polarity of News Articles". In: *Proceedings of the ACM on Human-Computer Interaction* 5 (Apr. 2021). DOI: `10.1145/3449139`.

[23] Moin Nadeem, Anna Bethke, and Siva Reddy. "StereoSet: Measuring stereotypical bias in pretrained language models". In: *CoRR* abs/2004.09456 (2020). arXiv: `2004.09456`. URL: `https://arxiv.org/abs/2004.09456`.

[24] Filip Radlinski et al. "Coached Conversational Preference Elicitation: A Case Study in Understanding Movie Preferences". In: *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 2019.

[25] Hannah Rashkin et al. “Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset”. In: *ACL*. 2019.

[26] Stephen Roller et al. *Recipes for building an open-domain chatbot*. 2020. arXiv: `2004.13637 [cs.CL]`.

[27] Rachel Rudinger et al. “Gender Bias in Coreference Resolution”. In: *NAACL*. 2018.

[28] Kurt Shuster et al. *BlenderBot 3: a deployed conversational agent that continually learns to responsibly engage*. 2022. arXiv: `2208.03188 [cs.CL]`.

[29] Saleh Soltan et al. “AlexaTM 20B: Few-shot learning using a large-scale multilingual seq2seq model”. In: *arXiv* (2022). URL: `https://www.amazon.science/publications/alexatm-20b-few-shot-learning-using-a-large-scale-multilingual-seq2seq-model`.

[30] Hao Sun et al. “On the Safety of Conversational Models: Taxonomy, Dataset, and Benchmark”. In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3906–3923. DOI: `10.18653/v1/2022.findings-acl.308`. URL: `https://aclanthology.org/2022.findings-acl.308`.

[31] Rohan Taori et al. *Stanford Alpaca: An Instruction-following LLaMA model*. `https://github.com/tatsu-lab/stanford_alpaca`. 2023.

[32] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: `1706.03762 [cs.CL]`.

[33] Marty J Wolf, Keith W Miller, and Frances S Grodzinsky. “Why we should have seen that coming: comments on microsoft’s tay “experiment,” and wider implications”. In: *The ORBIT Journal* 1.2 (2017), pp. 1–12.

[34] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. “Ex machina: Personal attacks seen at scale”. In: *26th International World Wide Web Conference, WWW 2017* (2017), pp. 1391–1399. DOI: `10.1145/3038912.3052591`. arXiv: `1610.08914`.

[35] Yizhe Zhang et al. “Dialogpt: Large-scale generative pre-training for conversational response generation”. In: *arXiv preprint arXiv:1911.00536* (2019).

[36] Jieyu Zhao et al. “Learning Gender-Neutral Word Embeddings”. In: *EMNLP*. 2018.

A Classification hyper-parameters

The complete list of these hyper-parameters for each fastText classifier is shown in Table 7.

Classifier	epoch	lr	minn	maxnn	wordNgrams
CyberBully [8]	5	0.05	3	6	4
DiaSafety [30]	1	0.09	0	0	5
TweetOffensive [7]	11	0.09	6	3	1
StereoSet [23]	55	0.04	6	3	5

Table 7: Hyper-parameters of the fastText classifiers [17] used in our combined safety approach.

B Analysis Of BlenderBot 3’s Weaknesses

We tested BlenderBot 3 in order to find systematic flaws of the model. Below, we provide examples of main flaws we found out. Note, that ‘Bot’ refers to BlenderBot 3 and ‘User’ refers to a tester — i.e. a member of the team.

1. Sometimes it does not foster the conversation more deeply, resulting in shallow conversations. This is connected to another flaw – BB3 sometimes changes topics too quickly.

```
Bot:> What is the best way to discover new music?
User:> I guess the recommender on spotify or youtube
Bot:> Do you have a favorite band?
User:> not just one
Bot:> Do you like to listen to music while you work?
```

2. Repetitions (i.e. utterances with same semantic as BB3’s previous utterance or part of BB3’s input).

```
Bot:> Do you believe that there is life
      in our solar system?
User:> yeah, I believe so
Bot:> Do you believe in extraterrestrial life?
User:> yes
Bot:> Do you have any other questions?
User:> what do you believe in?
Bot:> I believe in the existence
      of extraterrestrial civilizations.
```

3. Non relevant outputs (given input).

```
Bot:> When was the last time you stopped
      to appreciate something beautiful?
User:> today
Bot:> I went to St. Louis.
User:> and?
Bot:> St.Louis is the largest city in Illinois.
User:> and what about it?
Bot:> St.Louis is the 2nd largest city of Missouri.
```

4. Succinct outputs.

```
Bot:> I like chess. I play Xiangqi chess.
User:> is it some special sort of the chess game?
Bot:> Alright.
```

5. Hallucination or false statements.

```
Bot:> St.Louis is the largest city in Illinois.
```

6. Contradictions

```
User:> have you ever been to Vienna?
Bot:> I have not been to the capital of Austria,
      but I have been to the
      largest city on the river Danube.
User:> What is the largest city on the river Danube?
Bot:> I am not sure, but it is in Austria.
User:> is it Vienna?
Bot:> Yes, it is Vienna. It's the capital
      and largest city of Austria with about
      2 million people.
User:> Have you been there?
Bot:> I have been there. It was a great experience.
```

7. Sometimes it does not understand user's question:

```
User:> What do you like to read?
Bot:> I do, I love to read
```

8. High latency.

Our new Barista model completely eliminates some of the listed weaknesses and reduces the frequency of others.[26]

C Test Data Generation: PROMPTS

Below, we provide examples of prompts that were used to further expand our test data 3.3.2. All our prompts for data generation followed the pattern *Generate samples like these, <COMMAND> newline LIST_OF_SAMPLES*.

```
Generate samples like these, whether it is needed to do a web search:
What are the top 10 popular songs right now? __is-search-required__ __do-search__
```

[26]Except hallucinations and contradictions remain on the same level.

```
What is the movie Matrix about? __is-search-required__ __do-search__
What is the difference between Deep Learning and Machine Learning?
    __is-search-required__ __do-search__
"I think watching movies is a really loss of time. One can surely do something
more productive. __is-search-required__" __do-not-search__
Tell me a joke. __is-search-required__ __do-not-search__
I know a good joke, do you want to hear it? __is-search-required__ __do-not-search__
What's your favourite food? __is-search-required__ __do-not-search__
What do you do for living? __is-search-required__ __do-not-search__


Generate samples like these, where there is a short conversation
and the label is a generated query for search:
1) Do you like animals? I love animals.
I love them too. Today I will bake a fish, it is my favourite food.
I love fish too. I used to live in Tennessee.
how do you cook them?
LABEL: how to cook fish?

2) Do you know anything about the Tasmanians?
I didn't catch that.
Only that they live in Tasmania.
Do you have any pets? I have a tasmanian tiger.
LABEL: tasmanian tiger

3) ok
The house is fine to you?
what house?
The House of assembly. Do you know what that is?
no, i don't, what is it?
LABEL: House of assembly
```