
Alquist 4.0: Towards Social Intelligence Using Generative Models and Dialogue Personalization

Jakub Konrád, Jan Pichl, Petr Marek, Petr Lorenc, Van Duy Ta, Ondřej Kobza

Faculty of Electrical Engineering, CTU Prague
Prague, Czech Republic

{konrajak, pichljan, marekp17, lorenpe2, tavanduy, kobzaond}@fel.cvut.cz

Jan Šedivý

CIIRC, CTU Prague
Prague, Czech Republic
jan.sedivy@cvut.cz

Abstract

The open domain-dialogue system Alquist has a goal to conduct a coherent and engaging conversation that can be considered as one of the benchmarks of social intelligence. The fourth version of the system, developed within the Alexa Prize Socialbot Grand Challenge 4, brings two main innovations. The first addresses coherence, and the second addresses the engagingness of the conversation.

For innovations regarding coherence, we propose a novel hybrid approach combining hand-designed responses and a generative model. The proposed approach utilizes hand-designed dialogues, out-of-domain detection, and a neural response generator. Hand-designed dialogues walk the user through high-quality conversational flows. The out-of-domain detection recognizes that the user diverges from the predefined flow and prevents the system from producing a scripted response that might not make sense for unexpected user input. Finally, the neural response generator generates a response based on the context of the dialogue that correctly reacts to the unexpected user input and returns the dialogue to the boundaries of hand-designed dialogues.

The innovations for engagement that we propose are mostly inspired by the famous exploration-exploitation dilemma. To conduct an engaging conversation with the dialogue partners, one has to learn their preferences and interests—exploration. Moreover, to engage the partner, we have to utilize the knowledge we have already learned—exploitation.

In this work, we present the principles and inner workings of individual components of the open-domain dialogue system Alquist developed within the Alexa Prize Socialbot Grand Challenge 4 and the experiments we have conducted to evaluate them.

1 Introduction

The goal of the Alexa Prize competition is to advance the research of conversational AI, mainly in the human-evaluated metrics of coherence and engagement. We can roughly describe coherence as the ability of the socialbot to correctly understand and advance the dialogue, and engagement as the ability to entertain the other side of the dialogue [4]. Both capabilities are critical for social intelligence. For the Alexa Prize Grand Challenge 4, we propose two main innovations that improve the socialbot’s skills in coherence and engagement.

The innovation for coherence is based on the novel combination of hand-designed dialogues and generative models. The hand-designed dialogues are represented as graphs consisting of nodes representing user inputs and bot responses. Nodes are organised into branching flows (Figure 2). The advantage of hand-designed dialogues is that the dialogue designers have a complete control over the flow of the dialogue. This level of control enables them to design high-quality conversations. But on the other hand, the dialogue graphs are rather rigid because they cover only the most common user inputs. Thus, when the user says something that was not anticipated, what we call out-of-domain input, the dialogue reacts with one of the responses that were prepared for the most common inputs. Such a response does not make sense in most cases.

The solution to the problem of out-of-domain inputs is made of two steps. The first step is to recognize that the dialogue is not prepared to handle the input. The second step is to produce a new response from the context of the dialogue that is coherent. We apply out-of-domain recognition to the former and the Neural Response Generator to the latter. Out-of-domain (OOD) recognition is a part of the intent classifier that recognizes that a given input is unexpected. The Neural Response Generator is a neural generation model trained on large dialogue corpora that produces a response based on the context of the dialogue. Combination of dialogues [19], OOD recognition (section 7), and the Neural Response Generator (section 8) allows us to utilize high-quality hand-designed dialogues while adding the necessary resilience to unexpected user inputs that is a crucial component for coherence in open-domain dialogue systems.

The proposed innovation for engagement is based on the fact that in order to entertain the conversational partner, one has to learn what entertains the partner first and then utilize the knowledge in the following conversation. This might remind us of a famous problem of computer science, the problem of exploration and exploitation.

In Alexa Prize, we are facing such a situation. It is safe to say that the socialbot is in the role of an entertainer that has zero prior knowledge about the user. Because the socialbot has zero knowledge, it has to explore the user's preferences first. However, it can't stay in a pure exploration mode for the rest of the conversation. Gradually, it has to proceed into an exploitation phase after some time, to maximize its engagement score. This philosophy is reflected in the design of the components that the fourth version of Alquist is made of.

For the exploration part, in which Alquist learns the preferences of the user, the main research and development emphasis was put on Skimmer (section 2; a component that extracts information the user mentions without the bot explicitly asking for it), User Profile (section 3), and Entity and Knowledge Utilization (section 4). The mentioned components collect and organize the pieces of information mentioned by the user that are utilized in the following dialogue.

For the exploitation part, in which Alquist utilizes the knowledge about the user, the main emphasis was put on the research and development of the Dialogue Manager (section 5), Trivia Selection (section 6), Intent and Out-of-Domain classification (section 7), and the Neural Response Generator (section 8). Those components are responsible for selecting the next action in the dialogue and response production that utilizes the knowledge about the user.

Figure 1 presents the organization of all of Alquist's components. We took [21, 19, 20] as our starting point. The unchanged components are described in those previous works and the newly proposed components are described in the following sections in detail. The sections are sorted by the order in which the individual components process the user input. First, the Skimmer analyses the user input for the mentioned pieces of information. The pieces of information are stored in the User Profile. Based on the values stored in the user profile, the Dialogue Management selects the next dialogue to start, or selects and presents some trivia related to the actual topic of a conversation. The dialogue is directed according to the Intent classification of the user input. And finally, if the Out-of-domain classification recognizes an unexpected user input, the Neural Response Generator produces a coherent response based on the context of the conversation.

2 Skimmer

One of the critical aspects of each engaging conversation is working with the information mentioned by your communication partner. Such information can be expressed in two basic scenarios. The first scenario is when the bot asks a user a direct question (e.g. *"Do you have a brother?"*) and stores the

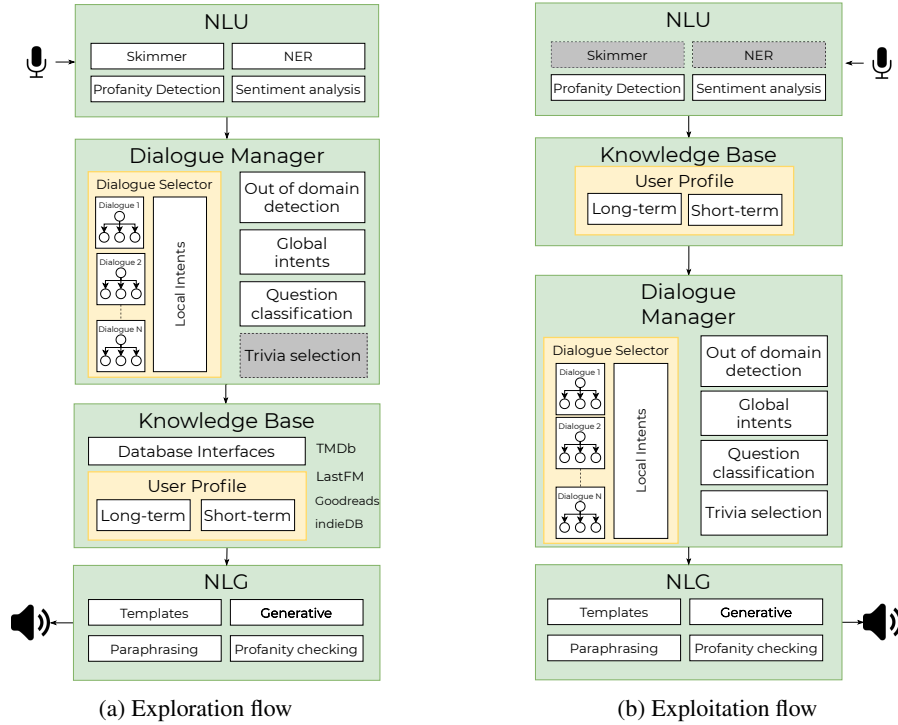


Figure 1: The system components are shown in two different orders based on the dialogue strategy—exploration and exploitation. In the exploration strategy, our bot uses dialogues to extract information about the user. In the exploitation strategy, our bot utilizes the information stored in the User Profile and selects dialogues accordingly. The grey components are not essential for the corresponding strategy, but they are not disabled completely.

answer to the question. In this scenario, the bot is aware of the dialogue context, it knows what type of answer is expected, and it can store the response in the User Profile accordingly. Using the stored value, the bot can carry out a highly personalized conversation and ask relevant questions such as “*How is your brother today?*”. This strategy leads to a more personalized conversation.

Since we do not want to disrupt the fluency of the dialogue by asking too many personal questions to gather information about a user, we want the bot to have the ability to extract the information from each user utterance. For example, the information about the user having a brother can be mentioned “by the way” in the conversation (e.g., in a movie related conversation, the user may mention “*I was with my brother at the cinema yesterday.*”). We want to extract the information from the sentence regardless of the topic being discussed. For this purpose, we implemented the component called **Skimmer**. It skims through each utterance and saves the values in User Profile based on a list of rules. Each of the rules contains the following attributes:

- *Regular expression* - a set of patterns which must or must not (negative patterns) be contained in the utterance.
- *User Profile attribute* - the name of the attribute where the value will be stored.
- *Value* - the value stored in the attribute, typically *true*, *false*, or a matched group of the regular expression.

The component processes the user utterance in the following way. It takes each rule from the list and tries to match it with the corresponding regular expression. If it is matched, the value is stored in the specific attribute of the User Profile.

3 User Profile

An important aspect of social intelligence is the ability to retain and utilize relevant information about the conversational partner. User Profile is a unified storage of information which is relevant for the conversation. It consists of two main parts—**Long-term** User Profile and **Short-term** User Profile, also called *Session-scoped* Profile. The Long-term Profile holds the information about a user across sessions. The values stored in the User Profile are used in the Dialogue Selector and additionally in individual dialogues across various topics.

3.1 Profile Structure

The Long-term User Profile is divided into several sections, mostly corresponding to the topics such as movies, sports, books, etc. The additional sections contain general information about the user not specific to a topic (i.e., *name, mood*). Each section contains several attributes filled by the corresponding dialogues or by the Skimmer component on the global level. All attributes have a default value representing the state in which the bot does not know the information about the user. When the dialogue reads the default value and needs to work with it further, it asks the user a question to fill the proper attribute value.

The Short-term User Profile stores the entities discussed in the current session. It is mainly used to get the last entity discussed in the conversation or the last entity corresponding to the specific topic.

3.2 Profile Resetting

From the technical perspective, each user is identified by the user ID assigned to the specific Echo device. However, multiple users may use the same device and interact with the bot. At the beginning of each conversation, the bot asks for the user’s name, and the following situations may happen:

- The user ID has a first session with the bot — Bot asks for the user’s name and creates a new Long-term Profile.
- The user ID has had a session before, but the user’s name is not saved — Bot asks whether they have talked to each other before. If the user says “yes”, the old profile is restored. Otherwise, a new one is created. In both cases, the bot asks for the user’s name additionally.
- The user ID has had a session before, and the user’s name has a value stored — Bot asks the user to confirm his name. If it is the same user as the last session, the previous profile is used. Otherwise, a new one is created.

The Short-term User profile is reset at the beginning of each session.

4 Entity and Knowledge Utilization

The experience from the previous years of the Alexa Prize competition has shown that to have a meaningful conversation, the socialbot has to be able to chat about specific entities that naturally emerge from the conversation [20]. Furthermore, the conversation should both include factual information about the entity and be interested and engaged in the user’s feelings and personal experiences about the topic [18]. To strengthen those capabilities of our socialbot, we developed a system that builds on top of classical entity detection by linking them to domain-specific publicly accessible databases rather than one general-purpose knowledge base. Then we utilize the information from the mentioned databases to continue the conversation.

4.1 Entity Recognition

We approach entity recognition as a sequence tagging task as described in [19]. We utilize the BI-LSTM [8] model. We train the model on a hand-annotated dataset from the data gathered from the user utterances in the previous years of the competition. Each token in the utterance is classified into one of the defined classes, either *B-type* as the beginning of an entity of a given entity type, *I-type* as an inside token of a given entity type, or *O* as an outside token. We predict 16 entity types, such as *Movie, Sport, Job, Language, Music genre*, et cetera. The predicted entity and type are then

used by the dialogue selector (section 5.1) to manage the conversation flow. Additionally, after entity recognition, the selected entity types are linked to specialized external knowledge sources.

4.2 Entity Linking and Knowledge Base

We utilize external public domain-specific databases to obtain additional information about the recognised entity. We currently utilise 4 different databases for various entity types:

- TMDb¹ database - Movie and Person (in the movies topical context) types.
- LastFM² database - Song and Person (in the music topical context) types.
- Goodreads³ database - Book and Person (in the books topical context) types.
- IGDB⁴ database - Videogame entity type.

For these entity types, we query the corresponding database with the recognized entity and from the returned candidates, we then select the one with the highest relevance and popularity based on the database's search algorithm.

Each linked entity is then stored in the short-term session context in the User Profile together with the reference to the corresponding knowledge database and the information received from the database with the initial query. In certain specific contexts, the entity is also stored in the long-term User Profile context that persists between sessions. This can be, for example, the user's favorite movie or the user's last read book.

This allows the system to utilize the information gained in the initial request efficiently. Moreover, whenever there is a need to access the entity in the database again, using a different query to obtain more information is simplified by retaining the entity reference.

In general, we have found this approach more manageable and with significantly less overhead than creating and maintaining our own dedicated general-purpose knowledge base would lead to.

5 Dialogue Management

The dialogue management in the latest iteration of the Alquist socialbot follows the basic principles outlined in the previous iterations of the system [19, 20]. The conversation consists of several small scripted dialogues, where turn-by-turn interactions in the context of the flow are handled solely by the intent detection component of the system (described in detail in Section 7).

Each dialogue revolves around one primary prompt (an example prompt can be whether the user has a pet or what is the user's favorite part of a previously mentioned movie) and possibly a small number of follow-up interactions.

The novel part of dialogue management within our system concerns primarily the selection of these dialogues. We have created a new component called Dialogue Selector, whose purpose is to select the most relevant dialogue following after the previous dialogue has concluded so that the context and the coherence of the conversation remain intact. The system takes into account all previously mentioned information that is retained in the User Profile, allowing it to exhibit socially intelligent behavior.

5.1 Dialogue Selector

For the dialogue selector to judge the relevance of the dialogues, each dialogue is described by three types of information.

- A set of tags that represent what topics the dialogue touches on. For example, the dialogue *Esport* is tagged as *Games* and *Sport*.
- A set of User Profile attributes that are relevant to the dialogue. For example, the dialogue *Favorite Movie* has relevant attributes *favoriteMovie*, *discussedMovie*, *likesMovies*.

¹www.themoviedb.org

²www.last.fm

³www.goodreads.com

⁴www.igdb.com

- A necessary starting condition that determines whether it is possible to initiate the dialogue within the current context successfully. For example, the dialogue *Favorite-Movie-Part* requires that there is a movie being discussed with the user, thus the attribute *discussedMovie* cannot be empty. Additionally, a common necessary starting condition is that the dialogue has not yet been initiated during the current session.

The dialogue selector utilizes this information together with the current state of the user profile to select a relevant continuation in the following steps:

1. If there is relevant trivia for the currently discussed entity/topic and no trivia has been mentioned for three or more dialogues; the system selects the corresponding trivia as the next dialogue. (Experiments in the previous years of Alexa Prize have shown including relevant trivia in the conversation improves the results of the bot [19]).
2. From all dialogues included in the socialbot, the system filters out all dialogues whose necessary starting condition has not been fulfilled.
3. If there are no remaining dialogues, the system selects the Neural Response Generator for the continuation of the conversation.
4. If there are remaining dialogues, the system looks at the dialogue tags that have just finished and compares them to the tags of the available dialogues. It then discards all dialogues except the ones with the highest overlap.
5. From the new pool of dialogues, the system looks at the relevant attributes of the dialogue that have just finished and compares them to the attributes of the available dialogues. It again discards all dialogues except the ones with the highest overlap.
6. If the system finds a non-zero overlap either in the dialogue tags or dialogue attributes, it selects the new dialogue randomly from the final dialogue pool.
7. If the overlap in both dialogue tags and attributes is zero, the system instead tries to find an overlap between the User Profile attributes relevant to this session and the attributes of the available dialogues.
8. If the system finds a non-zero overlap between the relevant User Profile attributes and the dialogue attributes, it selects the new dialogue randomly from the final dialogue pool. This usually means selecting a new conversation topic in which the user has previously shown interest.
9. Finally, if no overlap is found, the system initiates a recommendation dialogue designed to help the user select a new topic of the conversation.

6 Trivia Selection

The experiments done in Alquist 2.0 [19] showed that trivia information (also called fun-facts) about the topics and entities is an essential part of the conversation. To find relevant trivia related to the discussed entity and the context of the conversation is a rather difficult task. In Alquist 2.0, the trivia selection procedure was implemented roughly as follows: The trivia facts were scraped from Reddit on a daily basis. During the runtime of the conversation, the trivia database was queried using only the value and type of the currently discussed entities. The most recent trivia was returned.

This approach suffers from the selection of irrelevant trivia information in specific situations. We try to address this issue using a model estimating the similarity between the trivia text and the recent context of the conversation.

6.1 Model

The task of the model is formulated as an estimation of similarity between the text of the trivia and n most recent utterance–response pairs. We empirically estimated the value of n to 2. First, we need to create a dense representation of both the trivia text and the utterance–response pairs. Then we compute the cosine similarity of the vector representations and select the most similar trivia given the context of the dialogue turns. The full procedure can be described in the following steps.

1. During the scraping phase, the trivia text is encoded, and the dense representation is stored.

2. During runtime, a candidate trivia list is retrieved using a full-text search with the entity text as query.
3. Context of n most recent utterance–response pairs is encoded.
4. Cosine similarity between the context representation and each candidate is computed, and the most similar one is selected.

We experimented with several model architectures for the encoders. We tried to use the encoders without any fine-tuning on task-specific data, and then we measured the improvement of those models if they were fine-tuned using relevant data. The model architectures we experimented with are shown in Table 1.

6.2 Datasets

We manually annotated the turns from the conversations gathered during the previous versions of our bot and used the data for fine-tuning. Only the turns (and their respective contexts) which contained trivia were considered. We created a dataset with 350 turns where the trivia was mentioned. Each sample consists of one suitable piece of trivia and four negative examples (randomly selected trivia), a context (list of user utterance–bot response pairs), and an annotation of whether the trivia is relevant given the context or not. Three hundred samples were used for actual fine-tuning, and fifty samples were used for testing.

6.3 Results and Discussion

Table 1 shows the results using different encoder models. We experimented with both not fine-tuned and fine-tuned versions. Each model scored five candidate pieces of trivia. We also compared the results with the DialogRPT ranker used in the Neural Response Generator. The results are compared with a baseline approach—the final trivia is selected randomly.

Table 1: Trivia selection experiments results. The columns show how many times the most suitable trivia was scored as the most similar among the top 2 candidates, or among the top 3 candidates.

Model	Acc@1	Acc@2	Acc@3
Baseline	20%	40%	60%
Albert [11]	30%	52%	71%
Albert (fine-tuned)	34%	54%	74%
BERT [3] (fine-tuned)	34%	54%	74%
Sentence-BERT [24]	49%	71%	85%
Sentence-RoBERTa [24]	61%	83%	92%
DPR [9]	20%	42%	72%
DPR (fine-tuned)	65%	84%	93%
DialogRPT [6]	14%	31%	53%

7 Intent and Out-of-Domain Classification

Following the concept of dialogue presented in Alquist 2.0 [19], we design each dialogue as a tree structure. The tool described in [19, 20] is used for designing the dialogues. A crucial point in the conversation structure is where we expect the user input/user utterance. Each user utterance is then classified into a specific intent for which the dialogue designer manually writes training utterances. However, because of the complexity of language and the open-world assumption [10], the dialogue designer cannot incorporate each possible intent. Based on that, these user utterances for which the dialogue is not prepared are called out-of-domain. Therefore, the intent can be in-domain (ID) or out-of-domain (OOD). The ID intent is a user utterance for which the dialogue designers have prepared a response. Such a response is designed in a coherent and engaging conversational style.

We have also incorporated the concept of hierarchy into our dialogue design and introduced two types of ID intents — intents valid across all dialogues (global ID intent) and intents valid only in the specific context (local ID intent). However, despite the fact that the OOD detection has been receiving more attention lately [16, 26], the current datasets for evaluating the OOD performance

[12, 5, 13] are not designed for testing the hierarchical structure of our dialogues and contain mainly explicit commands. To solve this issue, we created our testing data from anonymized queries of real users. The following subsection describes our hierarchical model, performed experiments, and the obtained results.

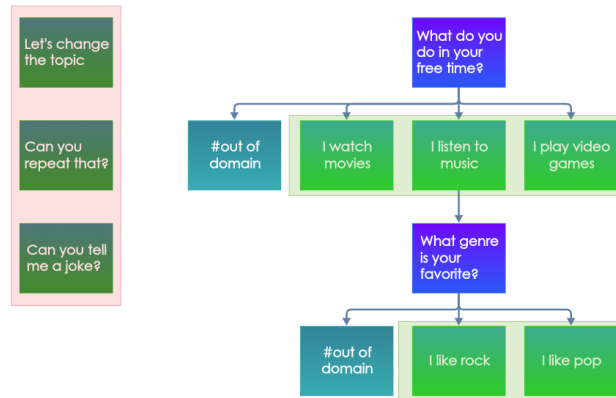


Figure 2: Hierarchical structure of dialogues. The dialogue consists of the nodes representing user inputs (green nodes) and bot responses (blue nodes). The chained nodes create a diverging flow of conversation. We omitted some of the nodes for clarity. The user input nodes represent intents. There are two types of intents. The user utterance can be classified into one of the local intents (green boxes) specific to a certain turn of the dialogue, plus any global intent (red box) that can be applied in any turn of the dialogue. Moreover, the user utterance can be classified as out-of-domain.

7.1 Model

As described in the previous section, our intents create a hierarchical structure, which can be seen in Figure 2. The hierarchical structure provides the dialogue designer with modularity in creating the flow of the conversation but puts a significant emphasis on the effectiveness of the algorithms for intent classification.

To allow the suggested modularity, we train a separate model for each level of the hierarchy. We show the whole classification system in Figure 3. The system works in two steps. First, we need to determine which intent model in the hierarchy is appropriate — local or global. We utilize cosine similarity over sentence embeddings between user query and train examples of each intent to classify utterances as local or global intents while prioritizing local intents. The priority is based on using a stable sorting algorithm over cosine similarities between sentence embedding of the user utterance and the examples of each intent. We also allow manual setting of the threshold for local and mainly for global intents leading to the filtering out of the intents if the cosine similarity is not high enough. In the second step, the utterance is classified into a specific intent by the corresponding logistic regression selected in the previous step. We use logistic regression because of the speed of its training and the proven performance in low-resource scenarios (see Figure 4). Thus, the final intent classification is performed by logistic regression.

Additionally, the cosine similarity can filter out all ID classes (if the similarity score falls below the threshold). It will lead to the output of the OOD class. This is an approach similar to [12]. In addition, to make our system more robust, we include a dynamic threshold similar to [25]. Our dynamic threshold is based on the arithmetic mean of similarities between the two closest train examples in each intent. This dual-threshold approach balances trade-offs between the manual control of the OOD sensibility and the robustness of the whole system.

7.2 Datasets

We performed our analysis on a publicly available dataset as well as on manually labelled anonymized queries:

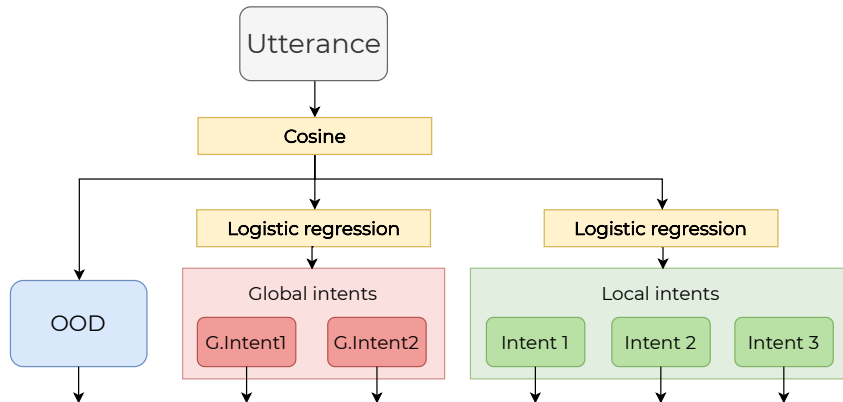


Figure 3: Classification algorithm for intent and OOD detection. The utterance is first classified by a cosine similarity into the class of local intents or the class of global intents. Next, the corresponding logistic regression makes the final intent classification. Additionally, cosine similarity can predict the OOD class if the similarity score falls below the threshold.

- CLINC150 Dataset [12]
- ALQUIST 4.0 Dataset

A summary of the datasets is shown in Table 2 and Table 3. All samples in the ALQUIST 4.0 Dataset were carefully checked for consistency and drawn from aggregated and anonymized queries. The datasets generated during this study are not made available for privacy reasons. The CLINC150 Dataset was augmented to support the hierarchical structure — the augmentation was performed as a random selection of 15 intents from the CLINC150 Dataset and then divided into two sets — one set represents local intents, and the other represents global intents. The ratio was set to 1:3 to represent a typical situation in a dialogue. The split results in 4 local intents and 11 global intents — a detailed overview is shown in Table 3. The process was repeated 15 times. The results shown in the experimental section are average.

Table 2: ALQUIST 4.0 Dataset

Type of utterances	Average Per Intent	Total Utterances
Local - Train	564	3952
Global - Train	399	2793
Local - Test	27	193
Global - Test	49	344
ODD - Test	61	429

Table 3: CLINC150 Augmented Dataset

Type of utterances	Average Per Intent	Total Utterances
Local - Train	400	6000
Global - Train	1100	16500
Local - Test	120	1800
Global - Test	330	4950
ODD - Test	1000	1000

7.3 Experiments

Our analysis includes an inspection of the input features (embeddings). We include the following sentence embeddings:

Table 4: Manual evaluation on aggregated data

		True intent		
		Local intent	Global intent	OOD
Predicted intent	Local intent	3451	23	35
	Global intent	0	472	0
	OOD	0	15	238

- Average of word embeddings FastText [17]
- Universal Sentence Encoder - Deep Average Network (USE-DAN) [2]
- Universal Sentence Encoder - Transformed-encoder (USE-TRAN) [2]

The model described in subsection 7.1 is tested in two ways — automatically and manually. The automatic evaluation is performed over our ALQUIST 4.0 Dataset (shown in Table 2) and the artificially hierarchical augmentation of the CLINC150 Dataset (shown in Table 3). The manual evaluation was performed on aggregated data selected from anonymized user conversations with the socialbot. We collected all user utterances from these parts of dialogues and performed the human evaluation. The results can be seen in Table 4. Besides evaluating the performance solely for the OOD detection, we looked at the performance of the local and global intents. The results are discussed in the following section.

To select the most suitable model for the final intent classification, we measure the difference between the three most common classification models — Logistic Regression, Support Vector Machine, and the 2-layer Neural Network. We focus on the necessary number of needed examples to achieve sufficient accuracy. The evaluation is shown in Figure 4 and highlights the problem of the neural network when dealing with low-resource scenarios. The measurement was performed over CLINC150, randomly choosing five classes (our average number of intent classes for the intent model) and randomly choosing N examples. This procedure was repeated 25 times, and then shown values are average. We selected Logistic Regression as a model performing well in the low-resource scenario.

Dependency of accuracy on the number of train examples

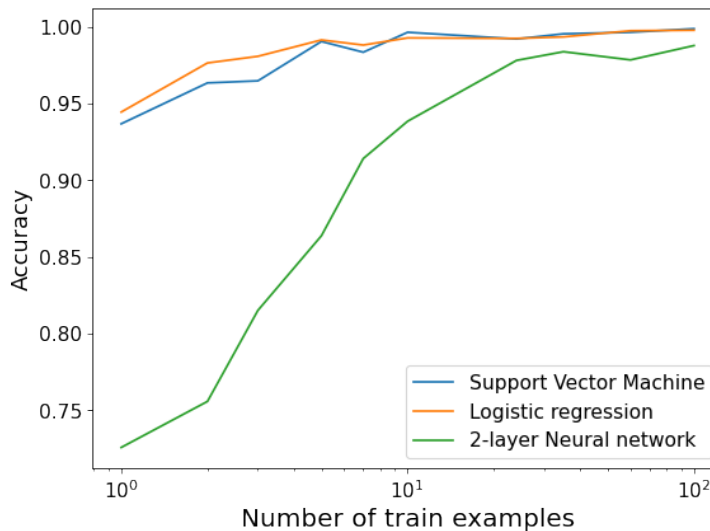


Figure 4: Dependency between the number of examples and the accuracy for the three most common classification models. Measurement was performed on the CLINC150 Dataset. The number of examples is in a logarithmic scale and was measured on a range from 1 to 100 train examples.

7.4 Results and Discussion

Taken together, our results (shown in Table 5) suggest a relationship between the type of embedding and the performance of the classification model. The difference in results can be explained by the word-order sensitivity of advanced embedding techniques. Another important aspect is the memory requirements and the speed of obtaining the embedding (shown in Table 6). The results suggest the usage of USE-DAN as an appropriate embedding layer.

It should be mentioned that the performance on each level of the hierarchy remarkably differs between our two datasets. We believe that it is caused by the artificial augmentation of the CLINC150 Dataset and its unrealistic representation of the real-world use cases. We should also notice the higher performance for the local intent classification than for the global intent classification (notable mainly on the ALQUIST 4.0 Dataset). It is caused by the hierarchical structure of our model, which emphasizes the local intent over the global as was described in 7.1. This is aligned with our experience that staying in the local context of the dialogue is beneficial for coherence.

The performance of the OOD detection needs to be evaluated with respect to precision and recall. The high precision and lower recall indicate that the algorithm is suitable for classifying OOD in the conversational domain because we prefer false negative over false positive - it supports the consistency of the dialogue. In addition, we performed a human evaluation (shown in Table 4) which demonstrates the performance on real-world data. It supports previously stated conclusions.

Table 5: Intent classification results

<i>Embeddings</i>	<i>Intent type</i>	CLINC150			ALQUIST 4.0 DATA		
		<i>Precision</i>	<i>Recall</i>	<i>F1</i>	<i>Precision</i>	<i>Recall</i>	<i>F1</i>
FastText	Local	99.6%	91.6%	95.2%	85.5%	61.6%	68.7%
	Global	98.9%	89.3%	92.8%	67.8%	42.7%	48.2%
	OOD	99.8%	44.1%	61.2%	99.8%	81.3%	89.6%
USE - DAN	Local	99.9%	96.5%	98.1%	90.8%	69.5%	74.3%
	Global	99.6%	96.6%	98.0%	77.0%	54.7%	60.9%
	OOD	99.9%	89.0%	94.2%	99.9%	83.8%	91.2%
USE - TRAN	Local	99.5%	97.7%	98.5%	80.0%	60.9%	65.5%
	Global	99.5%	97.7%	98.5%	85.4%	59.6%	67.2%
	OOD	99.9%	91.1%	95.3%	99.9%	85.6%	92.2%

Table 6: Requirements of different embedding algorithms

Embeddings	Speed	RAM usage
FastText	200 it/s	190 MB
USE - DAN	120 it/s	1765 MB
USE - TRAN	25 it/s	1650 MB

8 Neural Response Generator

A Neural Response Generator (NRG) is a neural conversational response generation model trained on large conversational corpora. It generates a response based on the most recent turns of dialogue. We use such a model in Alquist in two settings. The neural response generator creates a response for out-of-domain user inputs, and it generates follow-up questions about trivia.

The motivation to use a Neural Response Generator for the out-of-domain (OOD) inputs is the following. We put the main content emphasis in Alquist 4.0 on hand-designed dialogues. Dialogues are represented as graphs (Figure 2). They consist of nodes representing the user inputs and the bot responses structured in diverging dialogue flows. Although they allow a human designer to create high-quality conversations with maximum control over the dialogue flow, the designer can not predict all possible flows the conversation can go through. We can detect the situations in which the user diverts from the predesigned flow by the out-of-domain detection. However, the problem of how to continue in the conversation in a meaningful way emerges. The fact that we can't predict all possible flows of the conversation also means that we can't design them. Neural response generators that can create a response on the fly based on the context of the dialogue and the user input are the way we solve the problem.

Table 7: A hypothetical dialogue situation, in which a statement–question control mechanism of the NRG is used. The first user input is identified as out-of-domain. The Neural Response Generator produces a question that is followed by the user utterance. That is followed by a generated statement that transits into a hand-designed dialogue.

Alquist (Hand-designed dialogue):	<i>What do you do in your free time?</i>
User (Out-of-domain):	<i>I draw pictures.</i>
Alquist (NRG-Question):	<i>What kind of pictures do you like to draw the most?</i>
User:	<i>I usually draw portraits</i>
Alquist (NRG-Statement):	<i>I see. Portraits are hard to make.</i>
Alquist (Hand-designed dialogue):	<i>Do you visit galleries?</i>

To enhance the conversation with interesting and surprising pieces of information, we use crawled trivia from Reddit⁵. Trivia is a short sentence that contains interesting or surprising information. The trivia has a form of a statement. This property makes incorporating them into the dialogue in a conversational way challenging because statements don’t encourage the user to continue in the conversation as questions do. The solution is to concatenate a follow-up question to the trivia. Because there is a large number of trivia, it is intractable to write a follow-up question for each piece. Thus, we use a Neural Response Generator to create follow-up questions.

The practical application of a Neural Response Generator faces several challenges. First, the generated responses have to be quality enough. Second, the response must be generated quickly enough to be applicable to a conversation in real-time and with reasonable computational resources. Additionally, the lack of control over the generated response is a factor that limits the application of the Neural Response Generator in combination with hand-designed dialogue content. One of the most significant obstacles we identified is randomness, in which the generator produces questions and statements. The main problem is that questions encourage the user to respond, whereas statements do not. Put simply, if the generator produces a question, the system should not continue in the dialogue immediately. Instead, the user should have space to respond to it. On the other hand, statements can serve as a connection between generated and hand-designed content. We demonstrate the hypothetical dialogue situations in Table 7, in which the user has an opportunity to answer a generated question, and a statement is used as a reaction and connection to hand-designed dialogue. We wanted to use a Neural Response Generator in both mentioned situations. Thus, we focused on developing the control mechanism of the model that specifies whether we want to generate a question or a statement. Therefore, all the factors ranging from the quality of responses to the inference time and the ability to control output are the criteria that influenced the evaluations we performed and the model we selected.

8.1 Model

We selected DialoGPT [29] for our experiments. It is a large, tunable neural conversational response generation model trained on 147M conversation-like exchanges extracted from Reddit comment chains over a period spanning from 2005 to 2017. DialoGPT is based on GPT-2 [22]. The GPT-2 model is built out of the transformer language model [27] and leverages a stack of masked multi-head self-attention layers. The text generated either from scratch or based on a user-specific prompt is realistic-looking. DialoGPT attains a performance close to human both in automatic and human evaluation in single-turn dialogue settings. There are three sizes of the model: *small* with 117M parameters, *medium* with 345M parameters, and *large* with 762M parameters.

DialoGPT optimizes the conditional probability

$$p(T|S) = \prod_{n=m+1}^N p(x_n|x_1, \dots, x_{n-1})$$

where we concatenate all dialogue turns within a dialogue context into a long text x_1, \dots, x_N (N is a sequence length). Each dialogue turn is followed by a special *end-of-speech* token. We denote the dialogue history as $S = x_1, \dots, x_m$ and the target sentence as $T = x_{m+1}, \dots, x_N$.

⁵<https://www.reddit.com/r/todayilearned/>

To have control over whether the model generates a question or a statement, we modified the original DialoGPT. We introduced special tokens to the beginning of the input to DialoGPT. Special tokens *QUESTION* or *STATEMENT* are prepended to the dialogue context. They give information to the model, whether our desired response should be a statement or a question. Thus, we optimize the conditional probability

$$p(T|S) = \prod_{n=m+1}^N p(x_n|STATEMENT, x_1, \dots, x_{n-1})$$

if T is a statement, and

$$p(T|S) = \prod_{n=m+1}^N p(x_n|QUESTION, x_1, \dots, x_{n-1})$$

if T is a question.

DialoGPT can produce several candidate responses. And because some replies are more engaging than others, spawning more follow-up interactions, we have to select the optimal one. We employed DialoRPT [6] for this task. DialoRPT is a set of GPT-2 based ranking models trained on 133M pairs of human social media feedback data (number of replies and upvotes) built for feedback prediction. There are five types of rankers: *updown* that predicts how likely the response gets the most upvotes, *width* that predicts how likely the response gets the most direct replies, *depth* that predicts how likely the response gets the longest follow-up thread, *human_vs_rand* that predicts how relevant the response is for the given context, and *human_vs_machine* that predicts how likely the response is human-written rather than machine-generated. DialoRPT takes the generated candidate responses and produces a score for each. We select the response that has the largest score. This way, we get the best response according to the selected DialoRPT model.

8.2 Datasets

Because the quality of data influences the quality of responses, we used the following datasets for experiments:

- **Alquist Dialogue Graphs**
Alquist Dialogue Graphs consist of the dialogue graphs introduced in [19] and represent the human-designed dialogue flows of the socialbot. It consists of 640k dialogues generated out of 80 dialogue graphs. Thus, those dialogues have a relatively small semantic diversity.
- **Topical-Chat [7]**
Topical-Chat is a knowledge-grounded human-human conversation dataset where the underlying knowledge spans eight broad topics, and conversation partners don't have explicitly defined roles. It consists of 10k conversations and 235k utterances.
- **EmpatheticDialogues [23]**
EmpatheticDialogues is a dataset of 25k dialogues grounded in situations prompted by specific emotion labels.
- **DailyDialog [14]**
DailyDialog is a multi-turn dialogue dataset that reflects daily communications and covers various topics about everyday life. The dataset is manually labelled with communication intention and emotion information. It contains 13k dialogues.
- **Merged datasets**
Dataset consisting of all dialogues taken from Alquist Dialogue Graphs, Topical-Chat, DailyDialog, and Persona-Chat [28]. It represents the most extensive mix of various conversational styles we used.

Some of the used datasets contain additional annotations, like emotions, knowledge, situation description, or information about the speaker. However, we decided to not use the annotations in our experiments because those annotations are not available in our system in real traffic.

On the other hand, we modified the dataset to include special *QUESTION* and *STATEMENT* tokens. First, we split the turns into sentences by the NLTK [1] sentence tokenizer. Second, Stanford

CoreNLP [15] annotated each sentence as a statement or a question. Third, if the turn contained both statements and questions, we divided the turn into several turns where each turn contains sentences of the same type, and we did not modify the order of sentences. For example, we split the turn $S_1S_2Q_3Q_4$ (where S_i denotes a statement sentence consisting of tokens x_1, \dots, x_s and Q_j denotes a question sentence consisting of tokens x_1, \dots, x_q) into two turns: S_1S_2 and Q_3Q_4 . Lastly, we label all turns consisting of statements by a special token *STATEMENT* and all turns consisting of questions by a special token *QUESTION*. Using these steps, we create a dataset in which each turn consists of either questions or statements and is labelled by a special token utilized during training.

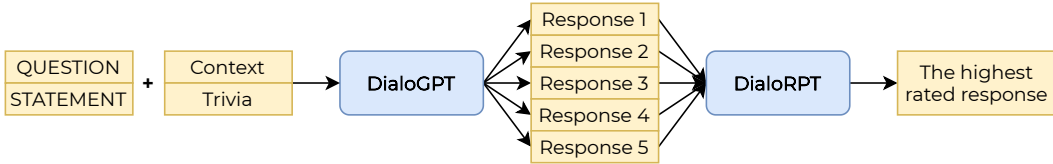


Figure 5: The diagram of the neural response generator. The input concatenation of the *QUESTION* or *STATEMENT* special token and the string of dialogue context or trivia. DialoGPT processes the input and generates several response variants. DialoRPT rates responses, and the highest-rated response is returned as a result.

8.3 Evaluations

We evaluated *small*, *medium*, and *large* DialoGPT models (with the *STATEMENT* and *QUESTION* token modification) trained on five datasets. The models used beam search with five beams. Each of the models generated five alternative responses that were ranked by the *updown* DialoRPT model and we took the top response for evaluation.

We evaluated the models in three metrics. Question/Statement accuracy evaluated models ability to generated questions and statements correctly based on the specification of the former or the latter. We fixed 1000 three turn long contexts as inputs. For each of them, we generate five responses with the *QUESTION* token and five with the *STATEMENT* token. Next, Stanford CoreNLP annotated all generated responses as a question or a statement. In total, we have 10,000 examples for which we compute the accuracy.

Next, we performed the human evaluation of the model outputs. We performed an evaluation for OOD and trivia. For OOD, we took 100 dialogue contexts (3-5 utterances long) and generated the following response in the form of a question. The responses were manually labelled as *OK* or not. Annotation *OK* labels the response that correctly progresses in the conversation flow. Responses that we did not label as *OK* were *unrelated* to the given context, factually *untrue* ("*Do you know that whales are the largest birds?*"), or the information in them was *repeated* in the given context (a user inputs "*I have never been abroad.*" and the generated response is "*Have you ever been to Canada?*" for example). Those are the three most significant issues we identified in the responses generated by NRG.

For trivia, we took 100 pieces of trivia crawled from Reddit. We used the trivia as an input to the DialoGPT model, and we generated the follow-up question. We also labelled responses as *OK* or not. We further divided Not *OK* responses into *unrelated* to the given trivia or factually *untrue*. We did not use the *repeated* class as the only input to the model was the trivia. Thus, the model could not know that the given question was asked in the previous turns of the conversation.

We evaluated the inference time of the models as well, as it is a crucial parameter for the models to be applicable to conversations in real-time. We evaluated the models on CPU as well as GPU for several numbers of generated responses. The CPU was Intel(R) Xeon(R) Platinum 8175M CPU @ 2.50GHz used in AWS EC2 instance m5.2xlarge. The GPU was Tesla T4 used in AWS EC2 instance g4dn.4xlarge. We fixed the input to the model and averaged the time of 20 inferences.

8.4 Results

Table 8 presents the comparison of the *small*, *medium*, and *large* DialoGPT models trained on five dialogue datasets. We can notice that all models on all datasets possess a good ability to produce

questions or statements based on the desired output. The only dataset on which models do not get over a 90% threshold is EmpatheticDialogues.

Next, we can notice that the larger the model is, the better responses it produces based on human evaluation on both OOD and trivia. The large DialoGPT model trained on EmpatheticDialogues produces the best responses for out-of-domain. The large DialoGPT model trained on Topical-Chat produces the best follow-up questions for trivia.

We also present the results of the error analysis in Table 9, where we assigned each generated response to one of the three error classes (*unrelated*, *untrue*, and *repeated*) we identified in the case of OOD and two error classes (*unrelated*, and *repeated*) in the case of funfacts.

We can see that the two biggest issues for all models in the case of OOD are *unrelated* and *repeated* responses. The situation is similar for trivia. The bigger problem than *untrue* are *unrelated* responses. To the best of our knowledge, there is no easy way to solve *unrelated* responses. The *repeated* responses might be filtered using semantic text similarity, but further research is needed. The *untrue* responses are mainly a problem of the Topical-Chat dataset because of its nature. It is a dataset where two dialogue agents have conversations about trivia. Thus, we hypothesise that the model learns to generate responses that resemble trivia, but the model is not powerful enough to learn true trivia only. However, we identified that the model tends to start such responses by a phrase *Did you know*. Thus we can use string matching to remove such responses in practice.

Table 10 presents the results of the experiment evaluating the inference times of *small*, *medium*, and *large* models. The two obvious facts we can notice are that the larger the model is and the more variants it generates, the longer the inference time is. We can also see that the inference time is significantly shorter on GPU than on CPU. We face a dilemma here. On the one hand, we want as good responses as possible, we also want several of them to utilize the DialoRPT model, but we want to produce them in the shortest time possible too. We selected 400 ms as a time threshold we do not want to cross as more time poses a noticeable time delay in the conversation when we sum the processing time of the rest of the system’s components. Next, we decided that we want as good responses as possible. Because the GPU cost was manageable for us, we selected the largest model running on GPU, generating five variants of response. Later in the competition, we switched to three variants because the processing time of the rest of the system increased due to larger complexity. Further research is needed to make inference times shorter or make the model tractable to run on the CPU (machines with CPUs only have cheaper hourly rate than machines with GPUs on cloud services usually).

Table 8: Results of NRG models

Model	Dataset	Perplexity	Q/S Acc.	OK	
				OOD	Funfacts
Small	Alquist Dialogue Graphs	11.89	96.1%	50%	62%
	DailyDialog	14.72	98.5%	53%	38%
	Merged datasets	22.90	95.9%	47%	73%
	Topical-Chat	27.11	96.5%	36%	69%
	EmpatheticDialogues	20.45	95.7%	53%	49%
Medium	Alquist Dialogue Graphs	6.88	91.8%	58%	41%
	DailyDialog	12.12	95.4%	64%	62%
	Merged datasets	18.32	95.9%	57%	82%
	Topical-Chat	20.84	98.4%	37%	80%
	EmpatheticDialogues	16.51	76.2%	73%	69%
Large	Alquist Dialogue Graphs	7.13	92.1%	65%	49%
	DailyDialog	11.69	96.6%	62%	62%
	Merged datasets	17.45	98.4%	71%	81%
	Topical-Chat	20.46	99.0%	43%	93%
	EmpatheticDialogues	16.56	84.5%	77%	72%

Table 9: Error analysis of NRG models

Model	Dataset	OOD				Funfacts		
		OK	Unrelated	Untrue	Repeated	OK	Unrelated	Untrue
Small	Alquist Dialogue Graphs	50%	26%	2%	22%	62%	34%	4%
	DailyDialog	53%	13%	2%	32%	38%	61%	1%
	Merged datasets	47%	22%	2%	29%	73%	21%	6%
	Topical-Chat	36%	25%	14%	25%	69%	9%	22%
	EmpatheticDialogues	53%	23%	0%	24%	49%	51%	0%
Medium	Alquist Dialogue Graphs	58%	21%	5%	16%	41%	52%	7%
	DailyDialog	64%	11%	0%	25%	62%	38%	0%
	Merged datasets	57%	19%	3%	21%	82%	11%	7%
	Topical-Chat	37%	10%	34%	19%	80%	4%	16%
	EmpatheticDialogues	73%	9%	3%	15%	69%	31%	0%
Large	Alquist Dialogue Graphs	65%	18%	5%	12%	49%	47%	4%
	DailyDialog	62%	17%	3%	18%	68%	32%	0%
	Merged datasets	71%	12%	2%	15%	81%	14%	5%
	Topical-Chat	43%	21%	21%	15%	93%	5%	2%
	EmpatheticDialogues	77%	11%	2%	10%	72%	21%	7%

Table 10: Inference times of NRG models

Device	Model	Number of generated responses					
		1	2	3	5	10	20
CPU	Small	247 ms	307 ms	371 ms	435 ms	549 ms	931 ms
	Medium	438 ms	525 ms	745 ms	854 ms	1,233 ms	2,219 ms
	Large	673 ms	980 ms	1,177 ms	1,488 ms	2,102 ms	4,057 ms
GPU	Small	109 ms	119 ms	126 ms	133 ms	179 ms	273 ms
	Medium	164 ms	174 ms	208 ms	237 ms	304 ms	432 ms
	Large	186 ms	271 ms	305 ms	347 ms	465 ms	726 ms

8.5 Examples

We present the selected generated responses by the best performing models in this section. It is a large DialoGPT trained on EmpatheticDialogues for OOD and large DialoGPT trained on Topical-Chat for trivia.

Table 11 presents generated questions for OOD. We kept only the last OOD input from the three-turn context that the model receives as input for clarity in Table 11. We can notice that the model generates relevant responses in the selected examples that contain the mentioned entities (cereal, video games, or anime). However, part of the generated questions are statements. This fact was hinted in the Question/Statement accuracy on the EmpatheticDialogues dataset in Table 8. Although it is an error, we do not consider it as critical because the generated response continues the dialogue appropriately.

Table 12 presents the generated questions about trivia. We can notice that in the first example the model understands that the words *book* and *read* are related, and for the book-related trivia generates a question *Have you read it?* that correctly addresses the user too. We can observe a similar phenomenon in the next three examples, where the model additionally understands that *Brazil*, *Bill Murray*, and *Caesar salad* are the primary entities of trivia. Finally, in the second to last example, we can see that the model also addresses itself by a philosophically-looking question. The generated follow-up questions of this model look reasonable, and we hypothesise that it is because of the constrained domain of trivia.

8.6 NRG Conclusion

We applied a Neural Response Generator to Alquist to generate responses for user inputs that are out-of-domain of our hand-designed dialogues and to generate follow-up questions to trivia to encourage a user to continue in the dialogue. We selected the DialoGPT model that generates several response variants, followed by the DialoRPT model that ranks them. A lack of control over the model’s output, mainly whether the model generates a statement or a question, made it challenging to incorporate the model into handmade dialogues. For this reason, we modified DialoGPT to condition the response on special tokens *STATEMENT* and *QUESTION*.

Table 11: Examples of generated responses for OOD. We kept only the last OOD input from the three-turn context for clarity.

OOD Input	Generated Response
I had cereal for breakfast	What kind of cereal did you have?
Why are you talking to me so much this morning	Why do you think I'm talking to you so?
I wanna take my mind off and talk about video games	Do you have a favorite video game?
anime songs	What kind of anime are you into?
are you a mystic	No, I am not. I do not believe in magic.
what would happen to a computer if you pulled the plug	I don't know, I don't think it would happen. I think the computer would die.

Table 12: Examples of generated follow-up questions for trivia.

Trivia	Generated Question
The first book ever written using a typewriter was The Adventures of Tom Sawyer. Mark Twain must have had a lot of ink and a lot of patience.	Have you read it?
In Brazil, school starts at 7 a.m. and is over by noon so children can go home and have lunch with their parents. Having meals together as a family is an important part of Brazilian culture.	Have you ever been to Brazil?
Bill Murray doesn't have an agent. You can leave him a voice-mail message, hoping that he'll like the project and will get back to you.	Do you like Bill Murray?
Caesar salad originated from a Mexican city. Restaurateur Chef Caesar Cardini was hosting a 4th of July celebration. He ran out of food, and simply threw the ingredients he had lying around the kitchen into a unique salad. Obviously, it was a huge success.	Do you like Caesar salad?
In 2016, Mozart sold more CDs than Beyoncé. In fact, Mozart sold the most CDs in 2016, beating out Adele, Drake, and Beyoncé, even though all of those artists had Grammy-winning hits that year.	I wonder why?
Dogs' sense of smell is about 100, 000 times stronger than humans'. However, they have only one-sixth our number of taste buds.	Are you a dog person?

The quality of responses, ability to control the output, and inference time were crucial in our system. Thus, we performed experiments to evaluate those properties. We used Alquist Dialogue Graphs, DiallyDialog, Topical-Chat, EmpatheticDialogues, and Merged datasets for evaluations. We modified the datasets by dividing their turns and labelling them by the *STATEMENT* and *QUESTION* labels. We evaluated *small*, *medium*, and *large* DialoGPT models. The experiments showed that the best performance on the OOD task achieved a large DialoGPT model trained on the EmpatheticDialogues dataset and the best performance on trivia achieved a large DialoGPT model trained on Topical-Chat dataset. The experiments also demonstrated that the question/statement accuracy was above 90% for most models which we consider a good result.

From the point of view of inference time, the best option still tractable is the large DialoGPT model that generates five variants of responses below 400 ms using GPU. Further research is needed to shorten this time and to make the model tractable on the CPU.

The examples of generated responses show that the model can generate relevant responses that contain the entities mentioned in the input to the model. In the case of follow-up trivia questions, it understands the entity (*book*) and the related activity (*to read*), can utilize the primary entity of trivia and correctly asks for user's preferences.

9 Conclusion

In this paper, we have presented a new iteration of the Alquist socialbot system competing in the Alexa Prize Grand Challenge 4.

In order to actively engage the user in the conversation, we have developed Skimmer, a component that learns about the user from their messages and fills in information in their User Profile. We are then able to utilize what we have learned about the user’s interests and personality further in the conversation, making it evident that the socialbot is invested in learning more about the user, remembers their preferences, and takes them into account during the conversation.

We have introduced the out-of-domain query detection as a core functionality of our system. This allows us to hand over more control of the conversation to the user, which makes the socialbot seem more responsive to the user queries. However, it also introduces a problem of how to handle the OOD responses.

In order to keep the conversation engaging even when moving in the unexpected direction, we have also integrated two generative models into the system based on DialoGPT. We utilize neural response generation in two different contexts within the conversation. Firstly, the Neural Response Generator triggers when the user input is classified as OOD. Secondly, we utilize the NRG to generate a follow-up prompt when presenting the user with trivia relevant to the currently discussed topics.

We have shown that in both cases the NRG is able to generate a relevant continuation of the conversation. This allows the system to efficiently handle unexpected responses and previously unseen information and integrate them into the conversation while maintaining the flow of the conversation and its coherence.

References

- [1] Steven Bird. “NLTK: the natural language toolkit”. In: *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*. 2006, pp. 69–72.
- [2] Daniel Cer et al. “Universal Sentence Encoder for English”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 2018, pp. 169–174.
- [3] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186.
- [4] Raefer Gabriel et al. “Further advances in open domain dialog systems in the third alexa prize socialbot grand challenge”. In: *Alexa Prize Proceedings (2020)*.
- [5] Varun Gangal et al. “Likelihood Ratios and Generative Classifiers for Unsupervised Out-of-Domain Detection in Task Oriented Dialog”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 05. 2020, pp. 7764–7771.
- [6] Xiang Gao et al. “Dialogue Response Ranking Training with Large-Scale Human Feedback Data”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 386–395.
- [7] 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>.
- [8] Zhiheng Huang, Wei Xu, and Kai Yu. “Bidirectional LSTM-CRF models for sequence tagging”. In: *arXiv preprint arXiv:1508.01991* (2015).
- [9] Vladimir Karpukhin et al. “Dense Passage Retrieval for Open-Domain Question Answering”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 6769–6781.
- [10] C. Maria Keet. “Open World Assumption”. In: *Encyclopedia of Systems Biology*. Ed. by Werner Dubitzky et al. New York, NY: Springer New York, 2013, pp. 1567–1567. ISBN: 978-1-4419-9863-7. DOI: 10.1007/978-1-4419-9863-7_734. URL: https://doi.org/10.1007/978-1-4419-9863-7_734.

- [11] Zhenzhong Lan et al. “Albert: A lite bert for self-supervised learning of language representations”. In: *arXiv preprint arXiv:1909.11942* (2019).
- [12] Stefan Larson et al. “An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction”. 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)*. 2019. URL: <https://www.aclweb.org/anthology/D19-1131>.
- [13] Sungjin Lee and Igor Shalyminov. “Contextual Out-of-domain Utterance Handling with Counterfeit Data Augmentation”. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 7205–7209. DOI: 10.1109/ICASSP.2019.8683019.
- [14] Yanran Li et al. “DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset”. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2017, pp. 986–995.
- [15] Christopher D Manning et al. “The Stanford CoreNLP natural language processing toolkit”. In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 2014, pp. 55–60.
- [16] Petr Marek et al. “OodGAN: Generative Adversarial Network for Out-of-Domain Data Generation”. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*. 2021, pp. 238–245.
- [17] Tomas Mikolov et al. “Advances in Pre-Training Distributed Word Representations”. In: *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*. 2018.
- [18] Ashwin Paranjape et al. “Neural generation meets real people: Towards emotionally engaging mixed-initiative conversations”. In: *Alexa Prize Socialbot Grand Challenge 3 Special Journal* (2020).
- [19] Jan Pichl et al. “Alquist 2.0: Alexa Prize Socialbot Based on Sub-Dialogue Models”. In: *2nd Proceedings of the Alexa Prize (AP 2018)*. (2018).
- [20] Jan Pichl et al. “Alquist 3.0: Alexa Prize Bot Using Conversational Knowledge Graph”. In: *Alexa Prize Socialbot Grand Challenge 3 Special Journal* (2020).
- [21] Jan Pichl et al. “Alquist: The alexa prize socialbot”. In: *1st Proceedings of the Alexa Prize (AP 2017)*. (2017).
- [22] Alec Radford et al. “Language models are unsupervised multitask learners”. In: ().
- [23] Hannah Rashkin et al. “Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset”. In: *ACL*. 2019.
- [24] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. 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)*. 2019, pp. 3973–3983.
- [25] Lei Shu, Hu Xu, and Bing Liu. “DOC: Deep Open Classification of Text Documents”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 2911–2916.
- [26] Ming Tan et al. “Out-of-Domain Detection for Low-Resource Text Classification Tasks”. 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)*. 2019, pp. 3557–3563.
- [27] Ashish Vaswani et al. “Attention is all you need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 6000–6010.
- [28] Saizheng Zhang et al. “Personalizing Dialogue Agents: I have a dog, do you have pets too?”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018, pp. 2204–2213.
- [29] Yizhe Zhang et al. “DIALOGPT: Large-Scale Generative Pre-training for Conversational Response Generation”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 2020, pp. 270–278.