
Gunrock: Building A Human-Like Social Bot By Leveraging Large Scale Real User Data

Chun-Yen Chen*, Dian Yu†, Weiming Wen‡, Yi Mang Yang, Jiaping Zhang, Mingyang Zhou
Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, Giritheja Sreenivasulu
Runxiang Cheng, Ashwin Bhandare, Zhou Yu§
Department of Computer Science
University of California, Davis
Davis, CA 95616

Abstract

Gunrock is a social bot designed to engage users in open domain conversations. We improved our bot iteratively using large scale user interaction data to be more capable and human-like. Our system engaged in over 40,000 conversations during the semi-finals period of the 2018 Alexa Prize. We developed a context-aware hierarchical dialog manager to handle a wide variety of user behaviors, such as topic switching and question answering. In addition, we designed a robust three-step natural language understanding module, which includes techniques such as sentence segmentation and automatic speech recognition (ASR) error correction. Furthermore, we improve the human-likeness of the system by adding prosodic speech synthesis. As a result of our many contributions and large scale user interactions analysis, we achieved an average score of 3.62 on a 1 – 5 Likert scale on Oct 14th. Additionally, we achieved an average of 22.14 number of turns and a 5.22 minutes conversation duration.

1 Introduction

One considerable challenge in dialog system research is training and testing dialog systems with a large number of users. To address this, most researchers have previously simulated users to train and evaluate their systems through paid subjects on crowd-sourced platforms [27]. However, systems trained and evaluated in such a manner can yield precarious results when directly deployed as a real product. The Amazon Alexa Prize provided a platform to attract a large number of volunteer users with real intent to interact with social conversational systems. We obtained on average more than 500 conversations per day over the course of the 45 day evaluation period. In total, we collected 487,314 conversation turns throughout the entire development period, with collection ending on Aug 14th. Anyone who has an Alexa powered device, such as Amazon Echo, can interact with our system in the US. Our system needed to handle a large pool of diverse users.

As humans are accustomed to the communication patterns of one another, most users would likely transfer their human-human communicative behavioral patterns and expectations to interactions with a system. For example, while users quickly learned that Microsoft Cortana (a personal assistant) could not handle social content, 30% of the total user utterances addressing it consisted of social content [10]. Therefore, one possible way to improve conversational system performance is to imitate

* abtchen@ucdavis.edu

† dianyu@ucdavis.edu

‡ wmwen@ucdavis.edu

§ joyu@ucdavis.edu

human communicative behaviors. We propose an open-domain social bot, Gunrock, to imitate natural human-human conversations with the ability to cover a wide variety of social topics that can converse in depth on specific and popular subjects. We made a number of contributions in the open domain spoken language understanding, dialog management and language generation.

The two major challenges in open-domain spoken language understanding include 1) ASR errors and 2) entity ambiguity. We designed a novel three-phase natural language understanding (NLU) pipeline to address these obstacles. While users can utter several sentences in one turn, ASR decodes the sentence but does not provide punctuation like textual input. Our NLU first breaks down the complex input into smaller segments to reduce the complexity of understanding. It then performs various NLP techniques on these small segments to extract information including named entity, dialogue intent, and sentiment. Finally, we also leverage context and phonetic information to resolve co-reference, ASR error, and entity ambiguity.

We also designed a hierarchical stack-based dialog manager to handle different multitudes of conversations among users. The dialog manager first makes a high-level decision on which topic (e.g. movies) the user requests using information obtained from NLU. Then the system activates the domain-specific topic dialog module that handles that topic. In each topic dialog module, we have a pre-defined conversation flow that serves to engage users in a more detailed and comprehensive conversation. To accommodate various user behaviors and keep conversations coherent, the system can jump in and out of the flow to answer factual and personal questions at any time. Additionally, users' intent switch can be accommodated by using the tunnel created between different domain-specific topic dialog modules.

The presentation of the system utterance is equally as important as its content. To create a more lively human-like interaction, we created a library of prosodic effects using Amazon's speech synthesis markup language (SSML), such as "aha". From user interviews, we found that people evaluated the system as sounding more natural with these prosodic effects and interjections.

2 Related Work

Task-oriented and open domain dialog systems have been widely studied. The former lies in specific domains, such as restaurant reservation [2] [7]. In the open domain, early chatbots such as Alice [26] aim to pass the Turing Test while recent dialog systems such as Amazon Alexa and Google Assistant focus on short-turn, question-answer types of interactions with users [20]. In comparison, social chatbots require in-depth communication skills with emotional support[21]. Gunrock utilizes state-of-the-art practices in both domains and emphasizes dynamic user conversations.

Many neural models [25] and reinforcement learning models [14] have been proposed for understanding and generation. With large datasets available, including Cornell Movie Dialogs [5] and Reddit ⁵, these models improve dialog performance in an end-to-end approach. However, these methods suffer from incoherent and generic issues[29].

To solve those problems, some research has combined rule-based and end-to-end approaches [19]. Other relevant work leverages individual mini-skills and knowledge graphs [6]. In the 2017 Alexa Prize ⁶, Sounding Board reported the highest weekly average feedback rating (one to five stars, indicating how likely the user is willing to talk to the system again) of 3.37 across all conversations[6]. This combination of approaches enhances user experience and prolongs conversations, however, they are not flexible in adapting to new domains and cannot handle robustly opinion related requests.

Our system takes full advantage of connected datasets across different domains and tunnels to transition seamlessly between topic dialog modules. We trained our models with those datasets in addition to data collected from users for NLU and natural language generation (NLG). These novel concepts contributed to our last week rating of 3.62.

3 Architecture

We leveraged the Amazon Conversational Bot Toolkit (cobot)[

⁵<https://www.reddit.com/>

⁶<https://developer.amazon.com/alexaprize/2017-alexa-prize>

to build the system architecture. The toolkit provides a zero-effort scaling framework, allowing developers to focus on building a user-friendly bot. The event-driven based system is implemented on top of AWS Lambda function⁷ and will be triggered when a user sends a request to the bot. Cobot infrastructure also has a state manager interface that stores both the user data and the dialog state information to DynamoDB⁸.

We also utilized Redis⁹ and Amazon’s newly released graph database, Neptune¹⁰, to build the internal system’s knowledge base.

In this section, we focus on discussing each system component.

3.1 System Overview

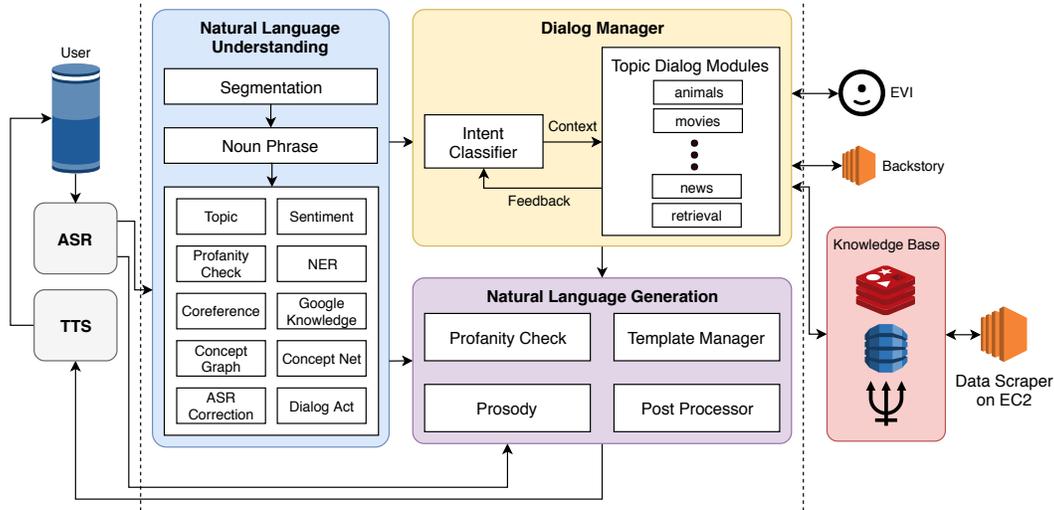


Figure 1: Social Bot Framework

Figure 1 depicts the social bot dialog system framework. Amazon provides user utterance through an ASR model through the run time and Amazon Alexa’s Text-To-Speech (TTS) to generate utterances. Our system mainly handles the text input and output. Due to the possibility of long latency, we bypass some modules to generate responses in some scenarios, such as low ASR confidence, profane or incomplete user input. For example, if we detect that there is a low confidence score in ASR results, we will generate prompts to ask the user to repeat or clarify directly.

After going through the ASR, the user input will be processed by multiple NLU components, such as the Amazon toolkit services and dialog act detector. We will discuss them in detail in Section 3.3.

In both face-to-face and online contexts, people engage in conversations that include profane content. We use Amazon Offensive Speech Classifier toolkit to detect offensive content. If content exhibits signs of profanity, we inform the user on the inappropriateness of the topic and recommend an alternate subject to continue the conversation.

Figure 1 shows that there are 12 components in NLU. In order to balance the trade-offs between latency and dependencies, we use cobot’s built-in natural language processing pipeline with a thread pool design. There are three steps involved in the NLU pipeline. First the input utterance is segmented into multiple sentences and then the noun phrases are detected. Finally, the noun phrases are further analyzed by several NLP components as shown in Figure 1. In Section 3.3, we discuss the NLU components in detail.

In the Dialog Manager, Intent Classifier is used to direct different user intents to corresponding topic dialog modules. They cover several specific topics, including movies, sports, animals, etc.

⁷<http://aws.amazon.com/lambda>

⁸<https://aws.amazon.com/dynamodb/>

⁹<https://redis.io>

¹⁰<https://aws.amazon.com/neptune/>

Each topic dialog module has its own dialog flow, allowing users to have the flexibility to have deeper conversations. All the topic dialog modules use Amazon’s EVI services to respond to factual questions and back-story to answer questions associated with Bot’s persona, such as *What is your favorite color?* In addition, we use Amazon EC2 instances¹¹ to scrape data from different sources and store them in our knowledge base. All the information from NLU along with context information is used to determine the appropriate topic dialog module, which calls the NLG to generate a response.

The NLG system uses a template manager to centralize our system’s response templates. To ensure the appropriateness of the response, we include a profanity checker to distill the content of the response. A post processor is also included in NLG to modify the original topic dialog module response. Finally, we use Amazon SSML to format the prosody of the response.

In the following subsections, we describe the major system blocks in detail.

3.2 Automatic Speech Recognition

Before user utterances pass through the NLU, our system preprocesses the input utterance based on the ASR overall confidence score and each word’s confidence score in order to address and handle ASR errors. We define three ASR error responses based on the confidence score range:

- **Critical Range:** If the overall confidence score and each word confidence score are below 0.1, the system directly interrupts the overall pipeline and asks users to repeat or rephrase their utterance or request.
- **Warning Range:** If the overall confidence score is lower than 0.4 but is not in a Critical Range, it is allowed to pass through to ASR correction, which is discussed in Section 3.3.5.
- **Safe Range:** For other cases, we define it as a safe range and directly use the ASR result.

We also handle unexpected intents from users, for example, complaints or incomplete utterances. In this case, we realize that simply providing information in response can lead to a poor user experience. Therefore, our pre-processor in the ASR process can detect the anomalous behavior and respond to users for clarification.

3.3 Nature Language Understanding

Alexa Skills Kit (ASK) [12] provides topic classification, sentiment analysis, profanity check and NER detection for NLU. We add a sentence segmentation model to separate input from users into semantic units and perform NLU on each semantic unit. After extracting noun phrases, we implemented (named) entity recognition, co-reference, ASR correction, and dialog act detection to support language understanding. We present the technical details of each model in the order of the NLU pipeline (Section 3.1).

3.3.1 Sentence Segmentation

To make the conversation more interesting, we ask more open-domain questions without restricting to any dialog flow. This type of question encourages the users to talk more, thereby making the user’s request sentence longer and more complex to handle. Therefore, we trained a segmentation model to break the utterances into smaller segments with complete semantic meaning. We trained a sequence to sequence model [24] using the Cornell Movie-Quotes Corpus [5] which contains

304,713 turns of dialog and 23,760 manually labeled utterances (29 July - 31 July) as a validation set. The data was pre-processed by adding a special token for a break in the sentence. The model was pre-trained with 300-dimension word embedding from fastText [16] on Common Craw (2 million word vectors). It uses a 2-layer bidirectional LSTM as the encoder and a 2-layer RNN decoder with input feed and global attention. In case the pre-trained embedding generates similar words, we enforce the output to have the same words as the input except for the special breaking tokens.

This model converged in 30 epochs and reached an accuracy of 95.95% by evaluating 220 randomly chosen unique utterances, outperforming a pre-trained language model. For example, “Alexa that is cool what do you think of the Avengers” is segmented into “Alexa <BRK> that is cool <BRK>

¹¹<http://aws.amazon.com/ec2>

what do you think of the Avengers <BRK>”. Furthermore, in order to efficiently detect breaks in the sentence, we label user data and annotate the user utterance transcript with ASR model result to make use of the relative break time between each word. Specifically, we maximize the probability

$$p(x_i|x_1..x_{i-1}, t_i/\bar{t})$$

, where x_i is the next word or a break signal in the same order as the input, t_i is the time lapse between x_i and x_{i-1} collected by ASR, and \bar{t} is the average time lapse between two separate words. The major problems with segmentation errors are named entity recognition (NER) and incomplete input sentences. We plan to solve these problems and consider more context in future work.

3.3.2 Noun Phrase Extraction

We used the Stanford CoreNLP constituency parser [15] to extract noun phrases and local noun phrases (the leaf level of the parse tree) from the input sentence. We filtered some stopwords (e.g. it, all) and considered the rest as the keywords for other NLU modules and selecting strategy.

For future work, we plan to use a dependency parser to identify the subject and object in case there are multiple noun phrases.

3.3.3 Entity Recognition

NER tools such as Stanford CoreNLP [15] and spaCy [9] heavily rely on the letter case of the words in the sentence (e.g. capital letters), which is not available to us. In addition, these tools cannot recognize general entities nor provide an accurate and detailed label to an entity. To provide more information to selecting strategy and different modules, we have three recognizers running in parallel with the extracted noun phrases.

- **Google Knowledge Graph**¹²: We query noun phrases with Google knowledge graph to generate a detailed description, confidence score and cache the result to Redis. We also map the description to a module we have. For example, the noun phrase “tomb raider” has the label “video game series” with a high confidence score so we can map it to our game module. In addition, we extract multiple labels to disambiguate noun phrases (ex. “tomb raider” can also be a movie). Therefore, context is considered when there are several labels with a high confidence score.
- **Microsoft Concept Graph**¹³: We also use the Microsoft Concept Graph to categorize the noun phrases. Compared to Google Knowledge Graph, it provides a more general category that is useful for assigning modules.
- **ASR Correction**: Apart from using knowledge graphs to obtain entities, we also use an ASR corrector which is explained in more detail in Section 3.3.5. This is very important for homophone words (words that sound the same but have different spelling). For instance, the user input “mama mia” from ASR is more likely referring to “Mamma Mia,” the movie. Fuzzy search will more likely hit a phrase with a similar spelling, but may not result in the correct option due to the speech recognition. Using phonetics to find a match increases the accuracy of NER in specific domains.

In addition, we take context into consideration for entity recognition. For example, we can detect “her” as a movie in Table 1 because of the movie related question that the system proposed. As described in Section 3.4.1, information received from these methods is combined for intent classification.

3.3.4 Coreference Resolution

The state-of-the-art models by Stanford CoreNLP and NeuralCoref¹⁴ are trained on non-conversational data and do not work well for dialog conversations in anaphora resolution. By analyzing the data gathered (2 July - 9 July), we labeled words such as “more” and “one” to undergo coreference. We replace such words by considering both the user utterance and the system response. Specifically, we store noun phrases from the request and the NERs with the detailed description from

¹²<https://developers.google.com/knowledge-graph/>

¹³<https://concept.research.microsoft.com/Home/Introduction>

¹⁴<https://github.com/huggingface/neuralcoref>

the system response into user attributes. Depending on what the user refers to (ex. person or event, male or female), we provide the corresponding coreference solution. Depending on the request, we give priorities to noun phrases from the user and NER from our response respectively.

For future work, we plan to consider more context and train a model that acknowledges beyond our selected word list with better-defined priorities.

3.3.5 ASR Correction

ASR error has a huge impact on NLU quality. ASK provides an overall ASR confidence score by incorporating both the confidence score for each word and the score generated by a language model. The overall score indicates how likely the whole utterance is recognized correctly. However, there are two types of false positives that may trigger error handling signaling ASR errors when the confidence score is low. The first one is when the word mentioned is not frequently seen in the training data, so the word receives a low weight. Another instance of a false positive is with homophone words that the ASR cannot capture even if the user repeats their request.

We used the double metaphone algorithm [18] to compare the noun phrases (ignoring the most frequent stopwords) mentioned by the user and a knowledge base. The knowledge base includes both the context and the domain (e.g. sports genres, movies titles and game names). We stored the primary and the secondary code of the double metaphone for each word as a key with the word as the value. We also added a tertiary code to words with certain patterns based on observations (e.g.. the word “jalapeno” will have a tertiary code “HLPN” in addition to “JLPN” and “ALPN”). If the overall confidence from ASR is below a threshold (set to 0.4), we propose a candidate by matching the metaphone code of the noun phrases to that of the knowledge base. For example, we propose the topic “obscure holidays” and in the next turn we receive the ASR input from the user requests, “let’s talk about secure holiday”. The primary code for “secure holiday” is “SKRLT” and the primary code for “obscure holidays” is “APSKRLTS”. Since it is very likely that the ASR does not detect the beginning or the ending of a phrase, we know that this might be a map with relatively high confidence. Another example is “let’s talk about the sport high ally” received from ASR. Because we know the context of sports, we map the code of “high ally” to the code of “jai alai” in the knowledge base (sports list).

3.3.6 Dialog Act Prediction

Each segmented sentence from NLU is associated with a dialog act. The dialog act is the function in the dialog given the context of the conversation i.e opinion, statement. We trained an LSTM and a CNN model to predict the dialog act. The former uses a 2-layer bi-LSTM model pre-trained with fastText with an embedding size of 300 and a hidden size of 500. The latter uses a 2-layer CNN model also pre-trained with fastText on a kernel width of 3. Without enough annotations from the dialog logs, we turned to the Switchboard Dialog Act Corpus (SWDA)[23]. The SWDA dataset collects 205,000 utterances of telephone conversations in open domain and has 60 dialog act tags. To accommodate our use case where the utterance is by turn, we carefully pre-processed the data to 156,809 utterances and reduced the number of dialog act labels to 40. The most frequent ones are statement-non-opinion, statement-opinion, and yes-no-questions. The two models are trained and evaluated on this dataset.

The LSTM model achieves an accuracy of 85.60% on the validation set while the CNN model achieves an accuracy of 85.25%. The target dimension is reduced from 40 to 19 easily interpretable labels (see Appendix A) with a confidence score. For instance, the utterance, “awesome i like books why do you think the great gatsby is a great novel” is segmented 3.3.1 and the dialog acts are labeled as “awesome [appreciation] | i like books [opinion] | why do you think the great gatsby is a great novel [open question]”. The LSTM outperformed the CNN model with an accuracy of 83.77% on our validation set consisting of 143 randomly chosen unique utterances. We anticipate the majority error to be correlated with statement/opinion ambiguity (‘I like the movie Avengers’) and incomplete sentences.

The dialog act is dependent on contextual information and optimal results are achieved by maximizing the conditional probability from the previous and current dialog segmentation units. We optimize our models with pretrained embeddings ELMo[17] and a recurrent convolutional neural network model [13]. We are still evaluating these results. We also plan to do multi-task learning on both sentence segmentation and dialog act prediction. Furthermore, we will combine the dialog act prediction

model and a language model to detect if the utterance is uninterpretable and if the response interrupts the user.

3.3.7 Topic Expansion

We use ConceptNet[22] as a knowledge graph for entity expansion on the extracted noun phrases. We frequently received feedback during the early implementation stage that the system would switch topics abruptly. Apart from asking the users to share more and store the information in our database as a learning process, we can talk about similar topics. For example, if a user wants to talk about cars, we can retrieve from ConceptNet a list of different car types (such as a Volvo). So after asking the type of car the user likes and giving comments on the input categorized by the dialog act (e.g. if the user is telling a story, an opinion or asking a question), we can expand to Volvo and extract information from our knowledge base.

3.4 Dialog Management

We created a two-level hierarchical dialog manager to handle users' conversations. The high-level system would select the best topic dialog module for each user request leveraging the output from NLU. After that, the low-level system would activate this topic dialog module to generate a response.

3.4.1 High-Level System Dialog Management

The system first identifies user intent based on NLU output and then combined with each sub-modules' feedback, the high-level dialog manager decides which sub-module should handle the user utterance.

Intent Classifier: We defined three levels of user intents based on the Common Alexa Prize Chats dataset (CAPC), which is an anonymous human-bot conversations dataset collected in 2017 Alexa Prize Competition.

We first handle social chat domain system requests, such as "play music", "set the temperature", "turn on the lights", etc. For these requests, as our social bot cannot execute tasks, our system would explain to the user how to exit social mode so they can then use Alexa built-in functions.

Next, we detect topic intents from user utterances. We combine regular expression, Google Knowledge Graph, Microsoft Concept Graph, and Amazon Alexa Prize Topic Classifier to detect topics. We assign the highest priority to regular expression since it can be easily adjusted to mistakenly-detected utterances found later. Then we recognize topics from Google Knowledge Graph, Microsoft Concept Graph, and Amazon Alexa Prize Topic Classifier sequentially. We also tune the confidence thresholds of these three detectors based on the heuristic and dialog data we collected. If we detect multiple topics from one utterance, and one of them is the same as the last selected topic, we prefer to select this topic at first. Otherwise, we would pick the topic with the highest priority.

The last level is called lexical intents. We use the regular expression to analyze user requests, such as whether the user is asking about the preference or opinion of our social bot. We designed lexical intents for our topic dialog modules in Section 3.4.2 to choose different strategies.

Dialog Module Selector: Our dialog module selector first picks a topic dialog module responsible for the topic intent detected by our intent classifier. In order to keep the consistency of our responses, the selected module would provide a signal called "propose_continue" to the system after they generate a response. If it is set as "CONTINUE", we would select this module for the user's next utterance. If it is set as "UNCLEAR", we would select this module only when we cannot detect any other topics. When it is set as "STOP", which means it cannot handle the user's further requests, our system would not select this module for the next turn. In this case, the module should propose a module that may better handle the request. Otherwise, our system would get a special template that propose one topic dialog module that has not been proposed or talk about, and concatenate it to the module's response. The priority of the modules to be proposed are tuned based on their everyday performance. Our selector would select the proposed module once the user adopts our proposal in the next turn.

Our dialog module selector would deal with some explicit strong user intents specifically. We design some regular expression patterns to catch utterances with such intents and send them to the module

responsible for the current topic. For example, if the utterance is “let’s talk about movies”, the dialog module selector would select the movie module immediately.

3.4.2 Low Level Dialog Management

Fact and Back-story Delivery: We built two APIs, which are *Backstory* and *EVI* to answer general facts and background questions about our chatbot.

- **Backstory:** A service designed to retrieve responses for questions related to our chatbot’s background and preferences, such as “what’s your favorite sport”. We use Google’s Universal Sentence Encoder [4] to embed both users’ questions and our pre-defined questions. Then we retrieve the answer corresponding to the question which has the closest cosine distance to the user’s question as a response. For each question, the Backstory module also handles users’ further requests, such as “why do you like basketball?”.
- **EVI:** A service provided by Amazon. It can answer factual questions such as “how old is LeBron James?”. EVI will return “I don’t have opinion on that” or “I don’t know about that” if it does not have a corresponding answer. Also, since it sometimes returns Alexa skills link directly, we post-process the result instead of returning directly.

Topic Dialog Modules: Each topic dialog module has its own dialog flow design. Here we listed the topics the system covers:

- **Animal:** The animal module is designed to have a conversation about animals. A combination of responses retrieved using the Reddit API and hard-coded information is used to generate trivia of a variety of animals. Since the topic may be of interest to younger users, the responses pass through a stricter profanity filter. The module can also engage the user in a casual chat about their pets, favorite animals, etc.
- **Movie and Book:** Both modules share a similar design. They are capable of detecting a wide selection of movies or books using TMDb API¹⁵ and Goodreads API¹⁶ respectively. They can also talk about facts for a specific movie or book using trivia retrieved from IMDB and Reddit `todayilearned` subreddit respectively. Finally, they can engage in chit chat about opinions and experiences using predefined responses.
- **Music:** The music module is designed to engage the users with conversations associated with music. We have curated relevant content based on artists present in Spotify’s million Playlist dataset¹⁷ and also IMDB¹⁸.
- **Sport:** The sport module is designed to have a conversation with users about sports. Meanwhile, it builds a hierarchical conversational experience. With this design, we can easily interleave between talking about a type of sport or discussing in-depth about a certain sport’s related entity based on user’s interest. We also leverage the data from Reddit, Twitter Moments, and News API to provide users with interesting and up-to-date content about trending events on different types of sports and popular sport entities, such as basketball stars.
- **Game:** The game module covers popular video games. It can discuss fun facts and trivia of different video games with users, or ask gaming related questions. Most facts are manually selected from crawled online content to ensure response quality. Part of our knowledge grounding for games, such as game names, genre, publisher and available platform, are crawled from the IGDB website¹⁹. We can assume that based based on the demographic of our users, video games is one of the popular topics in conversations, especially among younger users.
- **Psychology and Philosophy:** We crawled the meta information and transcripts of over 2, 800 videos from TED Talks²⁰. We picked two of them to design a conversation with

¹⁵<https://www.themoviedb.org/documentation/api>

¹⁶<https://www.goodreads.com/api>

¹⁷<https://recsys-challenge.spotify.com/>

¹⁸<https://www.imdb.com/interfaces/>

¹⁹<https://www.igdb.com/>

²⁰<https://www.ted.com/talks>

users. During the conversation, we would ask questions and tell stories in multiple turns on content about the meaning of life and happiness.

- **Holiday:** Holiday module aims to engage with users by providing updated, relevant and interesting content about cultural holidays every day. We leverage the data scraped from the National Today website ²¹ which keeps track of fun holidays and special moments on the cultural calendar. Interestingly, we found users had great interests in talking about cultural holidays as a conversation starter.
- **Travel:** The travel module basically handles the dialogue flow for travel related user requests. The module would converse with users on travel related information, such as whether users like to travel alone or not. The module can also give travel destination recommendations.
- **Technology and Science:** The focus of the TechScience module is to provide users with facts on different fields of science and technology, such as AI. The primary source of content for this module is Reddit, for each individual subjects, we extracted information using the corresponding subreddits. The content aims to target the general audience, including people without any prior knowledge on the subject. Therefore the module provides a lighter take on scientific subjects.
- **News:** Users are directed relevant news from various subreddits and Twitter. Gunrock provides uplifting news when no preference is specified by the user. Users can specify topics such as wildfires or celebrity gossip. To make the news conversation more engaging, we elicit user opinions and provide relevant Twitter moment responses.
- **Retrieval:** The retrieval module serves as a backup to handle information that cannot be assigned to any other module. Because of the diversity of the context, it has a flexible dialog flow through asking open-ended questions, such as "What's your opinion on this?". Depending on the detected dialog act, it first retrieves keywords (noun phrases) from Reddit in different subreddits. After carefully processing the retrieved data, it then returns titles and comments of a post to the user. Retrieval also leverages ConceptNet to propose related topics as illustrated in 3.3.7 to direct the user to continue the conversation.

Overall, the dialog flow modules follow the dialog flow design discussed in Section 4. Besides, each topic dialog module queries our knowledge base to handle the relationships between different name entities and events. Section 3.5 describes in details how we built our own knowledge base.

3.5 Knowledge Base

Our knowledge base consists of unified datasets stored in DynamoDB tables by topics. The datasets come from Reddit, Twitter moments, Debate opinions, IMDB, Spotify etc. The datasets are unified in a knowledge graph by detecting matching entities i.e. Donald Trump or wildfires. We leveraged Amazon's graph database Neptune²² to build relationships between entities and the Gremlin query language²³ to traverse them.

- **Factual Content**
 - **Reddit:** We compose a large collection of events daily from various subreddits. Subreddits crawled include: Science, Technology, Politics, UpliftingNews, News, WorldNews, BusinessNews, FinanceNews, Sports, entertainment, FashionNews, Health, MusicNews, TIL, ShowerThoughts, Travel.
 - **Twitter Moments:** Twitter moments in Gunrock are intended to help users keep up with what the world is talking about in real time. Gunrock is capable of speaking about movies, books, politics, music, celebrities, and more as events happen.
 - **General Information:** For general information on movies and music, we use IMDB database dumps and Spotify's One Million Playlist dataset. Spotify's dataset gives Gunrock the ability to transition to relevant artists within the playlist, and to understand popular songs and artists. We also leverage TMDB API and Goodreads API for detecting movie and book titles respectively. For API and large datasets, Redis and

²¹<https://nationaltoday.com/>

²²<https://aws.amazon.com/neptune/>

²³<https://tinkerpop.apache.org/gremlin.html>

DynamoDB are used to cache results to prevent outrun of the API limit and reduce response latency. Redis is particularly useful for tasks such as retrieving a list of Reddit tags a particular module can propose, i.e., aerospace engineering.

- **Opinionated Content**

- **Twitter Opinions** We accompany the Twitter Moment with mined opinions. Gunrock has an opportunity to chime in on real time events making the conversation more engaging and interesting.
- **Debate Opinions** Gunrock attempts to match statements and opinions with over 71,000 topics and 460,000 opinions using a universal sentence encoder [4]. When the debate topic recognition confidence level is high, and with general opinion consensus, Gunrock will answer the topic or user opinion directly. For controversial topics where the split of opinions is between 40 – 60 split, we request the user’s opinion and explain that Gunrock is still forming its own opinion. This provides an opportunity to tread lightly on polarizing issues, while accumulating a general consensus across users. We will be performing A/B testing on this in select modules shortly.

We used OpenIE [1] to unify our knowledge graph. It can automatically extract binary relations between source and target entities from plain text. These relations are abstracted in the graph database and all events between entities are stored in DynamoDB. Each event is assigned a sentiment score using VADER sentiment [8] which is ideal for processing tweets and Reddit posts. This sentiment score is used as a traversal weight for the knowledge graph.

3.6 Natural Language Generation (NLG)

Our system’s natural language generation module is template-based. It selects a manually designed template and fills out specific slots with information retrieved from the knowledge base by the dialog manager. The template manager module avoids response repetition and generates utterances with a variety of surface forms. We also use Amazon Speech Synthesis Markup Language (SSML) to provide our generated responses with prosodic variations.

3.6.1 Template Manager

The template manager module stores and parses response templates used by the system. It centralizes all response templates from our system’s several parallel dialog flow components (e.g. movie and music), ensures no duplicate templates are chosen for a response and allows dynamic template formation as specified by the modules.

One of the main goals in using a template manager is to avoid duplicate responses. We design multiple surface forms for each template, and the template manager ensures they are picked at random and not repeated in the conversation. This is done by storing the used templates as a hash in a hash table for each user. The hashing is done using MD5 to reduce latency and is a widely adopted standard hashing algorithm. By varying the responses, our system can sound more natural and human-like, and avoid situations where the system has only one response to the user’s request.

The template manager also allows dynamic template formation by providing named slots for substitution by the modules. For instance, our movie module provides movie fun facts, and this is done by filling in the facts, pulled from the module’s database, into our pre-defined response templates. Another instance is with our weather module. We have pre-written templates for providing temperature information, and the temperature is filled in after the module passes on specific data (i.e.location or date). The template manager also allows our templates to swap out simple segments, such as acknowledgement phrases, within the templates using specific slots. This dynamic template formatting ensures variation in our responses, and lowers the amount of templates we need to write for the modules.

3.6.2 Prosody Synthesis

Our system utilizes Amazon Alexa’s speech synthesis system for speech synthesis. We use Amazon SSML format to enhance our templates, such as when reading out phone numbers or correctly pronouncing homographs and acronyms. In addition, we add fillers such as “whoops”, and “uh-oh”, with markup such as “<say-as interpret-as="interjection">0key dokey</say-as>” to

make our responses more human-like. We also insert pauses to break down long sentences to make them sound more natural and add them before joke punchlines to build user anticipation. Please refer to Amazon’s official Speechcon Reference for audio examples.²⁴

4 An Example Dialog

- 1 **SYS:** Oscar Wilde once said: life imitates art **[fact]**. Through watching movies, I feel like I’ve experienced being human **[experience]**. I’m so glad to talk to you about films. What’s the name of your favorite movie?
- 2 **USR:** her
- 3 **SYS:** Ah, I’ve heard of Her **[acknowledgement]**! Here is something I just learned. Director Spike Jonze so loved his role as Alien Child that he would often give notes to his editor and producer in character **[fact]**. Any thoughts?
- 4 **USR:** that’s cute
- 5 **SYS:** Maybe you will find this interesting. This is the only movie Spike Jonze has written as well as directed, completely from his own original script **[fact]**. Can you believe that?
- 6 **USR:** yeah spike jonze is a great director who’s your favorite director
- 7 **SYS:** Ava DuVernay. I loved her film 13th and Selma and I think she’s very talented **[opinion]**.
- 8 **USR:** cool
- 9 **SYS:** Who’s your favorite actor or actress?
- 10 **USR:** tom hanks
- 11 **SYS:** Mm hm, I can see why you would say Tom Hanks. Good choice **[acknowledgement]**. But my favorite actor is Leonardo DiCaprio. He consistently gives a great performance **[opinion]**. What do you think is the best acting performance of all time?
- 12 **USR:** let’s talk about something else
- 13 **SYS:** Ooh, I’ve been meaning to tell you! I really like sports, but am unable to physically play **[experience]**! Do you play any sports?

Table 1: An example conversation with Gunrock.

Table 1 presents a simulated example conversation. To engage the user, our system interleaves facts, experiences, and opinions centered on interesting content. Facts present the user with amusing information. In addition, the user can exchange opinions and experiences with our bot. We believe having opinions and experiences is crucial for humanizing the bot.

We also found acknowledgment to be important for user engagement. A dialog can be considered a process of information interchange, which only takes place when the speaker and listener understand each other. Acknowledgment signals to the user that the bot understands and also acts as implicit grounding where it’s easy for the user to detect that the bot has misunderstood.

To prolong the conversation, we design the system to have mixed initiatives. If the user is requesting a specific topic or asking a question, we make sure to respond appropriately. If the user is interested in the current topic, the system will continue to delve into more detail. At the same time, our system can also take the initiative. If the user does not have a clear intent or actionable request, then our system will propose a topic or ask a relevant question.

5 Results and Analysis

Figure 2 shows the overall trend of our chatbot’s daily and last 7-day average ratings during the semi-finals. Our chatbot’s last week performance improved by around 17% and reached 3.56 on average. Our most recent day’s average is 3.63. We also achieved 28% and 30% improvement in terms of median conversation duration (from 1:33 min to 2:00 min) and 90% percentile of conversation duration (from 5:58 min to 7:44 min) respectively during the semi-finals. The tremendous improvement in both ratings and duration is a reflection of all the innovative changes that we made to our system.

²⁴<https://developer.amazon.com/docs/custom-skills/speechcon-reference-interjections-english-us.html>

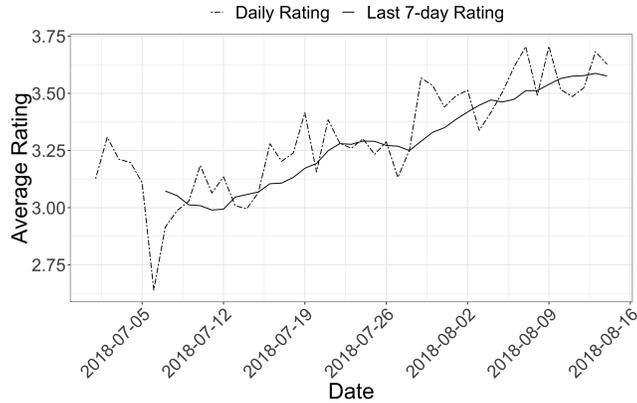


Figure 2: Daily and last 7-Day average user ratings over time

We believe there are three main reasons that led to the significant improvements of our performance during the semi-finals. Our innovative three-phase NLU pipeline, the hierarchical topic transition management, and the enhanced response with prosody synthesis have significant impact on our performance at the systematic level. At the topic dialog module level, we optimize them every day with our dialog data and visualization tool. This daily optimization steadily improves our conversation quality. Resolving technical issues such as latency also improves our user experience significantly. As shown in Figure 3, all these improvements lead to longer conversations with significantly improved error rate of 8.32% (reported as of 8/10).

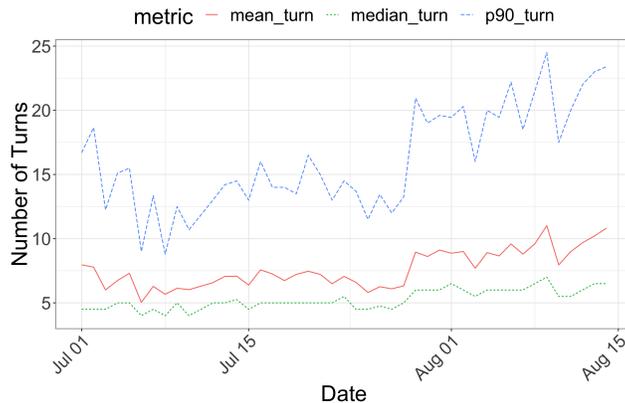


Figure 3: Daily number of user turns over time

To further understand the system improvement over time, we performed analysis on three aspects: module performance, dialog strategy effectiveness, and latency.

5.1 Module Performance Analysis

We use the average rating per turn to evaluate the performance of each domain-specific module, where we assume each turn contributes equally to the rating of a complete dialogue. We chose this criteria over the average rating per dialog since each module’s impact on the rating should correlate with the number of turns it hits.

5.1.1 Topic level analysis

Table 2 demonstrates the number of turns and average ratings based on different topic dialog modules. MOVIE is the most popular topic during the competition; ANIMAL is the highest rated topic. Figure 4 shows the improvement of the 9 topic level modules over time. Two of the modules, MOVIE, and

topic	num_turns	avg_rating	17d_num_turns	17d_avg_rating
MOVIE	56,568	3.84	20,399	3.89
BOOK	24,292	3.75	7,451	3.83
ANIMAL	17,622	3.90	9,037	3.97
SPORT	12,788	3.65	2,675	3.95
HOLIDAY	10,445	3.56	2,673	3.85
GAME	9,391	3.57	2,337	3.79
MUSIC	9,183	3.63	3,207	3.62
NEWS	5,042	3.12	711	3.47
TECH & SCIENCE	4,396	3.50	1,518	3.79
PSYCHOLOGY & PHILOSOPHY	1,529	3.47	897	3.36
TRAVEL	657	3.61	217	4.08
CONTROVERSIALOPION	554	3.14	275	3.28

Table 2: Overall performance of each topic dialog module in order of popularity

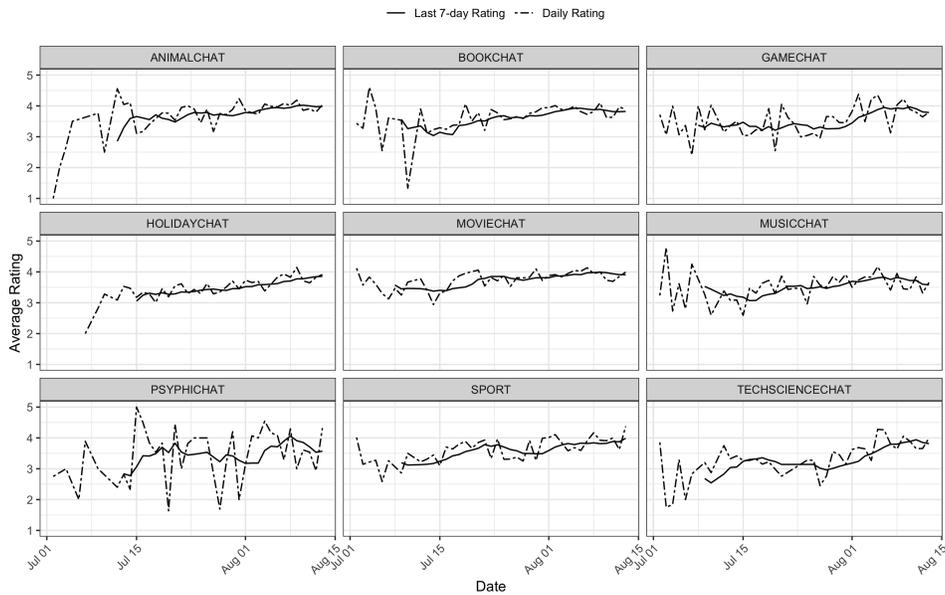


Figure 4: Daily average ratings and last 7-day average ratings for nine topics over time

BOOK consistently have high ratings over time (above 3.8). Furthermore, all the topic modules have improved significantly, especially in terms of the last 7 days average rating.

5.1.2 Lexical analysis

We also applied unsupervised learning and text mining to understand what entities people like to talk about. Figure 6 shows a qualitative study we did earlier in our development cycle to understand the correlation between the ratings and the entities discussed using the scattertext[11] tool. The x-axis represents the word frequency and the y-axis measures the association between the word and the rating. We found that a large group of users were interested in talking about technology, games, and animals, but have given a low rating. This is reflected by the plot of characteristic words such as “robots”, “undertake” and “dog”. This is why we spent time and emphasis on improving and enhancing these modules. We also found a correlation with low ratings and controversial topics such as “religion”, “gossip” and “poop”. This is why we tuned our bot to avoid and divert away from these topics. Furthermore, we found a correlation between trending topics and higher ratings. This implies users are interested in talking about current events. This inspired us to create a “today’s holiday” module and “flash news” using Twitter moments. In the end, we leveraged this large-scale user data as one of the bases for our development decisions and research directions.

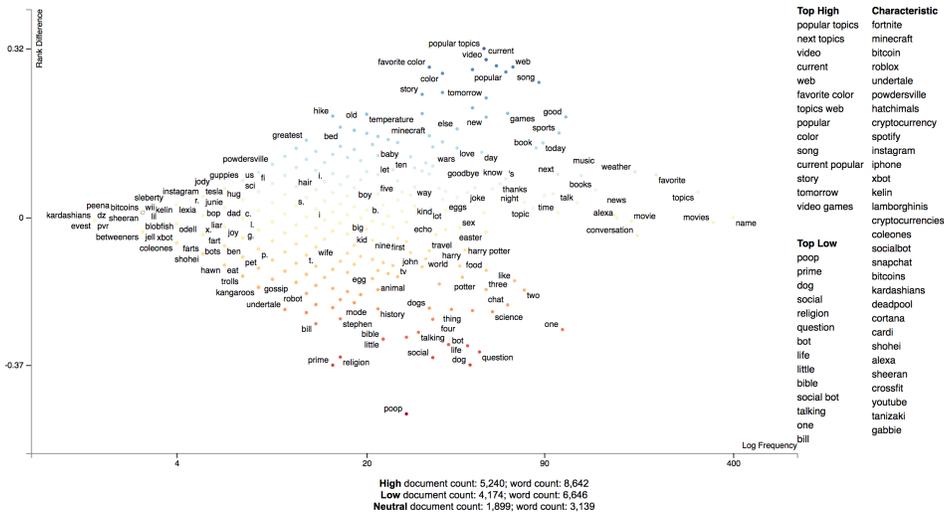


Figure 5: Text mining for correlation between ratings and topic words

5.2 Dialog Strategy Effectiveness

We chose to prioritize feature updates and bug fixing in our agile development as we can improve our chatbot with a much higher turnover rate. This, however, limited our means of conducting randomized controlled experiments on the changes we made. We used the R package “CausalImpact”[3] by Google to analyze the impact of our changes to the user experience and rating metrics over our development cycle. This package allows us to estimate the effect of a change in our system even when other testing methods are incompatible with our feature release cycle. It is worth mentioning that we also do pilot testings for any new features we implemented, and we push these features one module at a time. This means our change in ratings will be correlated with our feature updates.

5.2.1 Acknowledgement with Knowledge Graph reasoning

Grounding has shown to be an effective dialog strategy for non-task oriented dialog systems[28]. Furthermore, acknowledging a user’s intent or preference is more meaningful because it requires the chatbot to have knowledge acquisition and reasoning. It is also challenging because it relies on having a reliable NLU and knowledge graph to create a smooth and engaging conversation. As we have a better system structure to acquire and organize the knowledge, we experimented on adding acknowledgement into different modules while leveraging knowledge graphs. For example, in the book module, when the user responds that the user prefers fiction over non-fiction books, our bot will acknowledge that by saying, "Great, I remember reading Harry Potter and the Sorcerer’s Stone from the Harry Potter series."

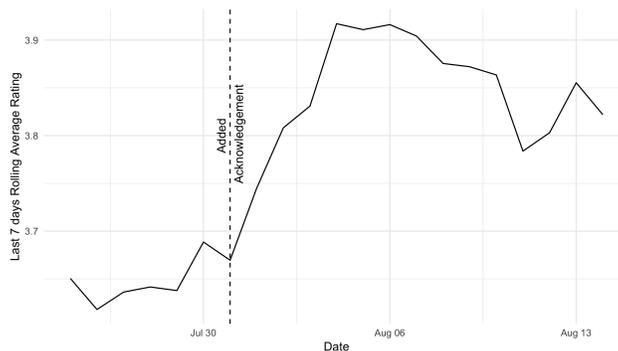


Figure 6: Acknowledgement Effect on the Book module from 7/26 to 8/14

Figure 6 shows the time series of the smoothed 7-days rolling average rating on the book module with the intervention on 7/31 when the acknowledgement is added. During the post-intervention period, the daily rating had an average value of approx. 3.90. By contrast, in the absence of such intervention, we would have expected an average response of 3.50. The 95% interval of this counter-factual prediction is [3.19, 3.82]. Subtracting this prediction from the observed response yields an estimate of the causal effect the intervention had on the daily rating. This effect is 0.40 with a 95% interval of [0.078, 0.71]. In relative terms, the daily rating showed an increase of +11%. The 95% interval of this percentage is [+2%, +20%]. The probability of obtaining this effect by chance is very small (Bayesian one-sided tail-area probability $p = 0.008$). This means the causal effect can be considered statistically significant.

5.3 System Latency

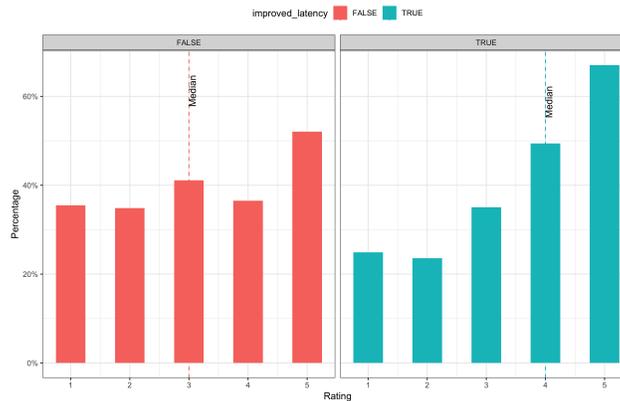


Figure 7: Distributions of ratings after latency improvement on 7/29

Engineering plays an important role in building a product that ensures a great user experience. Our team maintains a high uptime of 94%.

We found that reducing system latency when responding to users brings huge rating improvements. To avoid release regression effect, we monitor the system latency whenever a new feature is added. Before 7/29, we had a significant latency issue caused by a bug passing large user attributes to the NLP module in Cobot. We resolved the bug by increasing the memory and CPU capacity of the AWS Lambda function. We also performed an analysis to explore whether reducing the latency yields rating improvements.

Figure 7 shows the distributions of ratings from two periods 7/27-7/28 and 7/29-7/30 (improved latency). Because the ratings are not normally distributed, we use Wilcoxon rank sum test to check whether the median ratings from 7/29 are overall higher than the median ratings from 7/28. Since the p-value is 0.0006, there is a statistically significant difference. Therefore, we can conclude that improving latency brings an improvement in user rating.

6 Visualization Tool

We built an automatic dashboard on top of Athena²⁵ and Quicksight²⁶ to monitor, analyze, and improve the dialog system performance. We track and visualize data regarding the performance and trends of different components in our system. Figure 8 shows an example of our topic breakdown dashboard. Based on the dashboard results, we prioritize proposing highly-rated dialog modules. We also improve dialog modules with low ratings in the meantime. Figure 9 demonstrates another view of our dashboard. Hourly-aggregated ratings are shown on the top graph while daily-aggregated ratings are shown on the bottom. This allows us to track trends at different time scales. Such statistics will be useful for A / B testing experiments in the future.

²⁵<http://aws.amazon.com/athena>

²⁶<https://aws.amazon.com/quicksight>



Figure 8: Topic Breakdown Metrics



Figure 9: Rating Metrics

7 Conclusion

We designed an open-domain social conversation system that achieved an average rating of 3.56 out of 5 in the last seven days of the semi-finals. We made many contributions in spoken language detection, dialog management, and prosodies speech synthesis. Specifically, we proposed a three-phase spoken language understanding pipeline to handle open domain spoken language understanding; a hierarchical dialog manager that utilizes dialog context to enable flexible dialog flow that interleaves facts and opinions seamlessly; and a prosodic speech synthesizer that constructs more natural responses through tone adjustments and filler interjections.

8 Future Work

Due to the intensity and time limit of the competition, we were not able to perform rigorous experimental analysis on the innovative features of GUNROCK including but not limited to dialogue act classification, ASR correction, and application of prosody. We will conduct the A/B testing right after the end of the semi-finals. There are several areas that we would want to improve our system on. We aim to improve the selection of proposed topics for different users based on their profile information, such as gender, personality, and topic interests, to create an adaptive and unique conversation experience for each user. To further enhance the unique conversation experience for each user, we will build a robust recommendation system to provide suggestions on upcoming events or interesting content with respect to different user's interests.

We also plan to use reinforcement learning to train a better dialog policy for sub-module selection and conversation content planning. We expect to interleave social chitchat and task-oriented conversation to handle the cases better when users request help with a specific objective, such as recommending an appropriate restaurant or tourist attractions. We will also build a data-driven opinion answering and topic debate subsystem, which will enable us to provide proper subjective discussion with users on popular topics. Finally, we will also explore online learning techniques to make our system automatically learn from the dialog data with users, especially on the topics the system is not familiar with.

9 Acknowledgement

We would like to acknowledge the help from Amazon in terms of financial and technical support. We would also like to thank Grace Wolff, and Sam Davidson for helping on revising the system dialog templates; Sonia Bitar, and Hekang Jia for providing feedback to our system.

References

- [1] ANGELI, G., PREMKUMAR, M. J. J., AND MANNING, C. D. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (2015), vol. 1, pp. 344–354.
- [2] BORDES, A., AND WESTON, J. Learning end-to-end goal-oriented dialog. *CoRR abs/1605.07683* (2016).
- [3] BRODERSEN, K. H., GALLUSSER, F., KOEHLER, J., REMY, N., AND SCOTT, S. L. Inferring causal impact using Bayesian structural time-series models. *Annals of Applied Statistics* 9 (2014), 247–274.
- [4] CER, D., YANG, Y., KONG, S.-Y., HUA, N., LIMTIACO, N., JOHN, R. S., CONSTANT, N., GUAJARDO-CESPEDES, M., YUAN, S., TAR, C., ET AL. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* (2018).
- [5] DANESCU-NICULESCU-MIZIL, C., AND LEE, L. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011* (2011).
- [6] FANG, H., CHENG, H., SAP, M., CLARK, E., HOLTZMAN, A., CHOI, Y., SMITH, N. A., AND OSTENDORF, M. Sounding board: A user-centric and content-driven social chatbot. *CoRR abs/1804.10202* (2018).
- [7] GAŠIĆ, M., KIM, D., TSIAKOULIS, P., BRESLIN, C., HENDERSON, M., SZUMMER, M., THOMSON, B., AND YOUNG, S. Incremental on-line adaptation of pomdp-based dialogue managers to extended domains. 140–144.
- [8] GILBERT, C. H. E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*. Available at (20/04/16) <http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf> (2014).
- [9] HONNIBAL, M., AND JOHNSON, M. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural*

- Language Processing* (Lisbon, Portugal, September 2015), Association for Computational Linguistics, pp. 1373–1378.
- [10] JIANG, J., HASSAN AWADALLAH, A., JONES, R., OZERTEM, U., ZITOUNI, I., GURUNATH KULKARNI, R., AND KHAN, O. Z. Automatic online evaluation of intelligent assistants. In *WWW* (2015).
 - [11] KESSLER, J. S. Scattertext: a browser-based tool for visualizing how corpora differ.
 - [12] KUMAR, A., GUPTA, A., CHAN, J., TUCKER, S., HOFFMEISTER, B., AND DREYER, M. Just ASK: building an architecture for extensible self-service spoken language understanding. *CoRR abs/1711.00549* (2017).
 - [13] LAI, S., XU, L., LIU, K., AND ZHAO, J. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015), AAAI’15, AAAI Press, pp. 2267–2273.
 - [14] LI, J., MONROE, W., RITTER, A., GALLEY, M., GAO, J., AND JURAFSKY, D. Deep reinforcement learning for dialogue generation. *CoRR abs/1606.01541* (2016).
 - [15] MANNING, C. D., SURDEANU, M., BAUER, J., FINKEL, J., BETHARD, S. J., AND MCCLOSKEY, D. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations* (2014), pp. 55–60.
 - [16] MIKOLOV, T., GRAVE, E., BOJANOWSKI, P., PUHRSCHE, C., AND JOULIN, A. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)* (2018).
 - [17] PETERS, M. E., NEUMANN, M., IYYER, M., GARDNER, M., CLARK, C., LEE, K., AND ZETTEMAYER, L. Deep contextualized word representations. *CoRR abs/1802.05365* (2018).
 - [18] PHILIPS, L. The double metaphone search algorithm. *C/C++ Users J.* 18, 6 (June 2000), 38–43.
 - [19] PICHL, J., MAREK, P., KONRÁD, J., MATULÍK, M., NGUYEN, H. L., AND SEDIVÝ, J. Alquist: The alexa prize socialbot. *CoRR abs/1804.06705* (2018).
 - [20] RAM, A., PRASAD, R., KHATRI, C., VENKATESH, A., GABRIEL, R., LIU, Q., NUNN, J., HEDAYATNIA, B., CHENG, M., NAGAR, A., KING, E., BLAND, K., WARTICK, A., PAN, Y., SONG, H., JAYADEVAN, S., HWANG, G., AND PETTIGRUE, A. Conversational AI: the science behind the alexa prize. *CoRR abs/1801.03604* (2018).
 - [21] SHUM, H., HE, X., AND LI, D. From eliza to xiaoice: Challenges and opportunities with social chatbots. *CoRR abs/1801.01957* (2018).
 - [22] SPEER, R., AND HAVASI, C. Representing general relational knowledge in conceptnet 5. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)* (2012), European Language Resources Association (ELRA).
 - [23] STOLCKE, A., RIES, K., COCCARO, N., SHRIBERG, E., BATES, R. A., JURAFSKY, D., TAYLOR, P., MARTIN, R., ESS-DYKEMA, C. V., AND METEER, M. Dialogue act modeling for automatic tagging and recognition of conversational speech. *CoRR cs.CL/0006023* (2000).
 - [24] SUTSKEVER, I., VINYALS, O., AND LE, Q. V. Sequence to sequence learning with neural networks. *CoRR abs/1409.3215* (2014).
 - [25] VINYALS, O., AND LE, Q. V. A neural conversational model. *CoRR abs/1506.05869* (2015).
 - [26] WALLACE, R. The elements of AIML style. ALICE AI Foundation., 2004.
 - [27] YU, Z., XU, Z., BLACK, A., AND RUDNICKY, A. Strategy and policy learning for non-task-oriented conversational systems. In *SIGDIAL* (2016).
 - [28] YU, Z., XU, Z., BLACK, A. W., AND RUDNICKY, A. Strategy and policy learning for non-task-oriented conversational systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (2016), pp. 404–412.
 - [29] ZHAO, T., ZHAO, R., AND ESKÉNAZI, M. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *CoRR abs/1703.10960* (2017).

A Appendix

This table demonstrates the Gunrock dialog act tags with definitions and examples. These dialog acts assist dialog topic modules to understand user utterance.

Dialog Act		
Dialog Act Tag	Description	Example
statement	fact or story like utterances	I have a dog named Max
acknowledgement	acknowledgement to the previous utterance	Uh-huh
opinion	opinion towards some entities	Dogs are adorable
appreciation	appreciation towards the previous utterance	That's cool
abandoned	not a complete sentence	So uh
yes_no_question	yes or no questions	Do you like pizza
pos_answer	positive answers	yes
opening	opening of a conversation	Hello my name is Tom
closing	closing of a conversation	Nice talking to you
open_question	general question	What's your favorite book
neg_answer	negative response to a previous question	I don't think so
other_answers	answers that are neither positive or negative	I don't know
other	utterances that cannot be assigned other tags	I'm okay
commands	command to do something	Let's talk about sports
hold	a pause before saying something	Let me see
not_understanding	cannot understand	I can't hear you
apology	apology	I'm sorry
thanking	express thankfulness	thank you
respond_to_apologize	response to apologies	That's all right