Emersonbot: Information-Focused Conversational AI Emory University at the Alexa Prize 2017 Challenge

Zihao Wang
Department of Mathematics
Computer Science
Emory University
Atlanta, GA 30346
`zihao.wang2@emory.edu`

Ali Ahmadvand
Department of Mathematics
Computer Science
Emory University
Atlanta, GA 30346
`ali.ahmadvand@emory.edu`

Jason Ingyu Choi
Department of Mathematics
Computer Science
Emory University
Atlanta, GA 30346
`in.gyu.choi@emory.edu`

Payam Karisani
Department of Mathematics
Computer Science
Emory University
Atlanta, GA 30346
`payam.karisani@emory.edu`

Eugene Agichtein
Department of Mathematics
Computer Science
Emory University
Atlanta, GA 30346
`eugene.agichtein@emory.edu`

Abstract

We describe an information-oriented conversational AI system, EmersonBot, developed for the Alexa Prize 2017 competition. The main goal of the system was informing users about current events, and answering their questions, while maintaining a fluent conversation. The main innovations of Emersonbot include the development of a federated multi-source information retrieval system that is aware of the conversation context; a general dialogue management system which uses machine learning to analyze the user responses, complemented with rule-based conversational logic component, specifically tailored for the Alexa Prize competition rules. This report provides a detailed description of the system, and includes extensive analysis of the contributions of the various components to user satisfaction and engagement with the system. Our results show a steady and substantial improvement in user ratings and user engagement over the period of the semi-finals as the system was refined, especially in the last weeks as entity-oriented search and recommendation features were introduced. We conclude the report by outlining promising research directions, which could improve the satisfaction of the users, while also advancing the state of the art in conversation-oriented search.

1 Introduction

Open-ended conversational agent is a daunting challenge for current Artificial Intelligence techniques. We believe that Open-Domain Conversational Search (ODCS) is a crucial component in truly informative and engaging conversational AI agents, and is an important research direction in its own right. Unlike in traditional Web search or automatic question answering, which typically consists of one-sided interactions of a user making a request/question and the system responding with one or more answers, a conversation could allow the user to better formulate and understand their information needs, and even help them ask the right questions by proactively recommending information relevant to the conversation context.

Our entry in to the Alexa 2017 Conversational AI challenge, Emersonbot, focused primarily on developing the ODCS capability. Specifically, we aimed to retrieve requested information instantly

1st Proceedings of Alexa Prize (Alexa Prize 2017).

upon users' requests, and to proactively recommend information if the user does not know what to ask; to maintain the state, and flow, of conversation to allow the user to naturally explore topics of interest; and to automatically suggest new topics when appropriate.

Before introducing our system, we briefly overview previous work in conversational AI, to put our contribution in context. Along the way of developing conversation AIs, two categories of systems have been commonly explored. The first category is general chatbots, which are mainly designed for daily conversations. These chatbots usually generate answers by using preset scenarios or templates that match queries and answers[1], or uses machine learning techniques to learn various patterns and predict the future responses by using sequential decision methods[2]. These types of approaches perform great in terms of maintaining a natural conversational flow, but it is difficult to ensure that the information from generated responses are accurate due to limited templates and training data. The second category is information-retrieval based bots, focusing on techniques to retrieve related answers from real-time information database such as daily news[3]. However, because information retrieval tasks are independent, connecting separate queries into a coherent, human-like conversation becomes challenging. In addition, an open-domain system has always been a challenging task, especially for the design of conversation logic to maintain the flow of conversation.

The main goal of this project to provide an open-domain conversational search to inform and entertain users while retrieving related information. This task cannot be simply done by using one of the two aforementioned categories of systems, but rather by combining the two approaches organically. The main approach for Emersonbot is to use conversational-oriented search within nine different information retrieval components and connect all of them by designing a conversational logic. Emersonbot is divided into three major parts, which are hierarchical classification[12], sophisticated dialogue management and multiple retrieval components. After the classification process identifies users' intents, one of the retrieval components is chosen based on the conversational logic and retrieves the requested information simultaneously. The final response is returned according to the preset scenarios in our logic table. Modern NLP techniques, such as coreference resolution and entity recognition are also incorporated in order to make the system more intelligent. Various suggestion techniques, such as hot topic recommendations assist Emerson to continue the conversation within a topic that user prefers and improve the overall length and quality of conversations.

In the end of the report, extensive analyses on the performance of the system are conducted mainly based on the ratings of users and length of conversation between the system and Alexa users.

2 Approach

Emersonbot is built within the Flask framework as a web service. The current system is deployed on the Amazon EC2 server and connected to Alexa devices through a Lambda function. The flowchart of the connection between users and Emersonbot is shown in Figure 1.

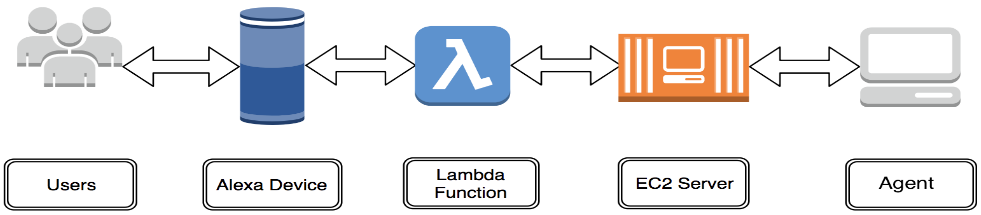

Figure 1: System deployment on Amazon EC2 servers.

In the above figure, users send requests to out agent through an Alexa device. A Lambda function connects the Alexa Skills Kit with our agent on EC2 servers.

In this section, the dialogue manager, which includes dialogue logging, conversation logic and conversation transition, and information retrieval components are introduced in details.

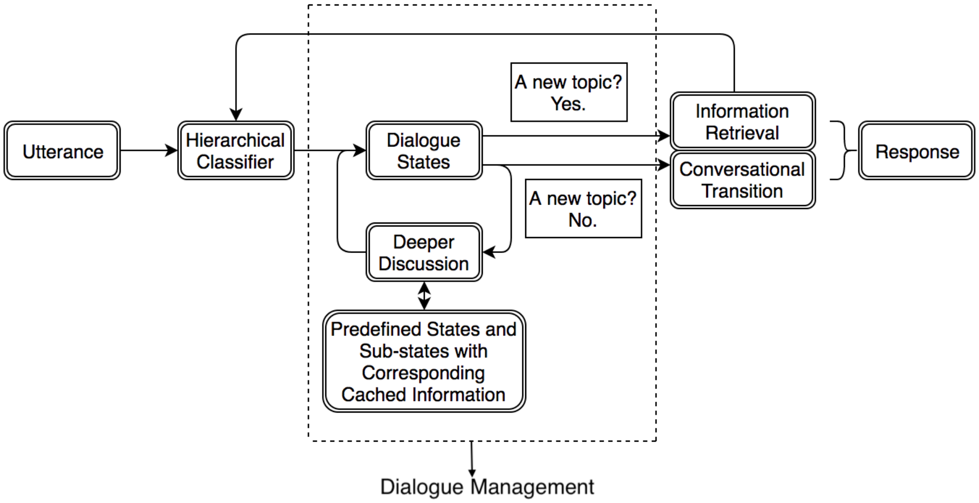

Figure 2: The Emersonbot dialogue manager.

2.1 Dialogue manager

The dialogue manager maintains the overall ecosystem. It keeps the system log, tracks conversation states, performs user intent classification, conducts natural language processing (NLP), checks information completeness, identifies named entities, directs utterances to designated modules, suggests related topics, and pinpoints response patterns. The detailed flow of conversation inside the dialogue manager is shown in Figure 2.1.

In the dialogue manager, the General Classifier performs a multi-class classification task. There are in total of 14 classes, which is shown in Table 3. The classifier is Gradient Boosting with Word2Vec and TF-IDF as features. The NLP processing is an integration of several techniques, which include named entity tagging[4], keyword extraction[5], string matching based on Levenshtein distance[6] and co-reference resolution[7]. The preprocessed texts are sent to the corresponding information retrieval modules to search for the proper responses. The information completeness check-up is based on special needs of certain modules. For example, the weather module needs location information to provide the weather information. If location information is not detected, follow-up questions will be asked to the users. The conversational transition is designed to have preset scenarios and logic, corresponding to different information retrieval modules, to assist the intent classification, as well as to enable the conversation to be more natural with preset response patterns and sample responses.

2.1.1 System logging

The system logging function is designed to store every bit of information through the conversation. This function logs three aspects of information, which are action, context, and utility.

- Action stores responses from each called component per each turn of conversation. Action is cleared before the next turn.
- Context logs all the information generated during the conversation through predefined variables. There are mainly three categories of variables, state variables, cache variables, and context variables. State variables are updated whenever the conversation state changes. Cache variables store intermediate information in one conversation state and are cleared when the state changes. Context variables store information related to utterances and responses. For example, when a user starts to chat about movies, a state variable is set as "Movies", a cache variable stores information extracted from Rotten Tomatoes, and a context variable stores all the utterances from that user.
- Utility deals with exception information, file system information, logging system information, and response time control.

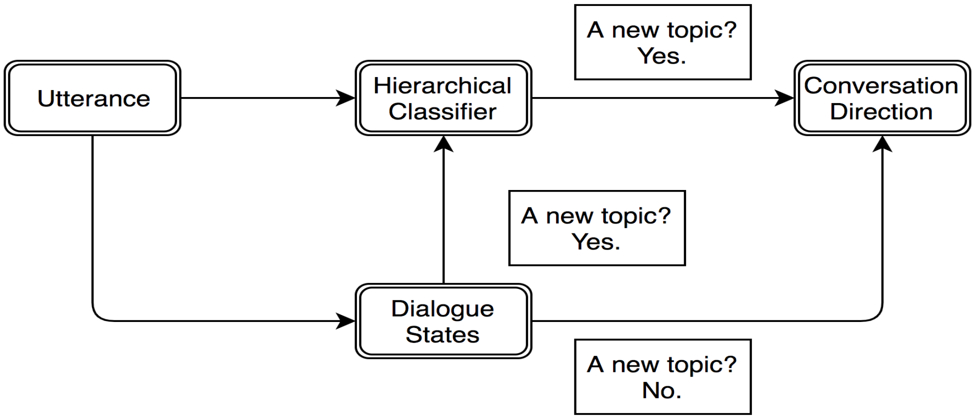

Figure 3: Conversation flow logic.

2.1.2 Conversation state tracking

Conversation state tracking is based on a design of conversation logic. Based on the utterances, the system first checks all the state variables, and matches the utterance with preset response patterns for this state. If there are no matches, the state changes and the classifier result directs the utterance to the corresponding module to retrieve an answer. Figure 3 illustrates the conversation logic.

2.1.3 Conversational transition

Conversation transition is a group of preset scenarios of response patterns. Whenever a conversation state is determined, a corresponding response pattern is determined to return to the user. The purpose is to provide a natural conversation experience and proactively drive the conversation.

2.2 Information retrieval components

There are in total 11 independent and 2 complimentary components. The 12 independent components are small-talk, jokes, music, movies, wiki-info, liveQA, weather, news, opinions, food, and default. The 2 complimentary components are wiki-how and hot-topics.

- Small-talk. The small-talk module is composed by AIML language based templates that gives standard responses to general conversations, like greetings, factoid information, or preset system information. The templates are modifications of Alicebot[8]. This component is mainly a transition function to direct the conversation to information retrieval components by giving a concise response and making a proposal of certain topics.
- News. The News module updates users with latest news in different categories, such as politics, sports, technology, etc. The sources of news come from WashingtonPost, CNN, and other mainstream media. News component returns the recent news based on category and keyword depending on how user asks.
- Wiki-info. This module is to inform people with related information from Wikipedia on entities of interest. The Wikipedia API is used to fulfill this function.
- Weather. The Weather module is used to provide general temperature and weather information, such as sunny, rainy and 90 degrees Fahrenheit. The source of this information is Yahoo! Weather. To support this feature, the module requests the user for city information during the conversation.
- Jokes. Jokes are for relaxation. When people feel bad at the greeting state or request for jokes, random jokes will be provided to cheer up the users.
- LiveQA. LiveQA handles various questions that cover nearly every possible aspect of life. Because the range of incoming questions are very broad, LiveQA component uses similarity matrix to find the best information from the Yahoo! Webscope Comprehensive Questions

and Answers. In order to increase the accuracy of response, Wikihow source is incorporated into liveQA to specifically deal with How-To questions.

- Movies. The Movies module collects movie information from IMDB and Rotten Tomatoes. General information about specific movies, such as actors, directors, plots and reviews can be extracted. Once entering the movie component, users will be guided into different paths of conversation. Users can be asked to choose from different aspects of a movie, Rotten Tomatoes box-office information, hot topic suggestions and component suggestions.
- Music. The Music module provides music recommendations and artist information from Billboard and relative news about corresponding singer, upcoming bands and concerts information. The information source is Billboard website. Similar to the movie state, music component can lead users to different paths of conversation.
- Opinions. The Opinions module currently returns a default answer to prohibit giving personal opinions.
- Food. The Food module mostly deals with recipe retrieval and food recipe recommendations. The food state will ask users to choose from aspects of recipe such as ingredients, reviews and directions once the recipe is retrieved. The source of recipes is epicurious.com website.
- Unrecognized. The unrecognized module is reached when classifier cannot recognize the users' intent. The module asks users to rephrase or ask a different question.

3 Experimental Setup

Emersonbot was integrated onto the Amazon Alexa platform to conduct converstaions with Alexa users, covering several topics, such as entertainment, fashion, politics, sports, and technology. Alexa users are encouraged to interact with the system and give feedback. All socialbots and Alexa users are anonymized during the contest to ensure the objectivity.

3.1 Performance evaluation metrics

3.1.1 Explicit user ratings

Alexa users are encouraged to give ratings, ranging from 1.0 to 5.0, after each conversation. 1.0 indicates unsatisfactory experience and 5.0 indicates satisfactory experience. This metric is considered as the primary metric for the agent performance evaluation.

3.1.2 Number of conversation turns

Each conversation contains one or more turns. A conversation turn is one utterance from a user and one response from the system. A complete conversation sample is shown in Table 1.

The number of conversation turns indicates how engaged users are with the socialbot overall. Therefore, it can be used as an evaluation metric for the system performance.

3.1.3 User engagement

Several components of the agent have the capacity to engage users in deeper discussions on a topic. The number of turns within a component can be used as a metric to evaluate the performance of one component, and the agent's overall performance combining all the components.

3.2 Users

The users came from two populations: Amazon Alexa team staff members and general Alexa users. With no demographic information and users spreading across the US, we assume that overall, users' opinions towards the performance of a socialbot are unbiased.

3.3 Experimental Period

The reported data were collected from June 15th to August 30th, 46 days in total.

Table 1: A sample conversation

Emersonbot: How are you?
User: i'm good thank you.
Emersonbot: Spectacular. I have something interesting about NBA, would you like to hear about it?
User: Sure.
Emersonbot: Here is sports news about NBA, JR Smith and Jason Day talk golf, hoops and 'running into' LeBron at a Cavs game. JR Smith and Jason Day sat down with ESPN.com to chat about the similarities between basketball and golf, Stephen Curry's professional debut and that time LeBron James accidentally crashed into Day's wife, Ellie, at a Cavs game back in 2015. Would you like another one?
User: No thanks.
Emersonbot: OK, would you like to talk about Music?
User: No thanks.
Emersonbot: OK. would you like to hear a joke, or would you like to talk about Movies, Food, News, Weather.
User: News.
Emersonbot: I have related news in the following categories, business, technology, entertainment, politics, sport. Which one do you want to know?
User: Politics is fine.
Emersonbot: Here is what I got from cnn on August 4th. President Trump today: Live updates . Trump is set to meet with his national security adviser, and full transcripts of his phone calls with Mexico and Australia have been released. Follow along here for updates. Would you like more news in this category?

3.4 Data collection

The results in the paper are based on over 22,000 session and 230,000 turns of conversations. Due to confidentiality agreement, all the sample conversations in this paper are regenerated by our researchers.

4 System analysis, results and discussion

The system analysis is based on the conversation transcripts and ratings from Alexa users. We started our analysis from logs after middle of June when the system became relatively stable. The focus of analysis is to compare the system performance before and after certain changes were made to the system, based on rating scores and the length of conversation (number of conversation turns). We analyze from three aspects, the effects of intent classification on the agent's performance, the effect of the proactiveness mechanism on the agent's performance, and the capacity of engagement within a component on the performance of the agent.

4.1 Effect of intent classification on the agent's performance

During the project, four major changes are made to the intent classifier.

1. Instead of using a classifier, an utterance was sent to all the components and a response from one component was finally picked based on a similarity score between an utterance and a response, calculated by cosine similarity with pre-trained word vectors as features. Since several components were able to give related information, the similarity score was not reliable to pick the desired answer.

2. Naive Bayes classifier was used to send an utterance to one specific component, but since it was a relatively simple classifier, the accuracy of classification was not satisfactory.

3. Changed Naive Bayes classifier to a Gradient Boosted Decision Tree (GBDT) classifier[9]. We analyzed the performance of GBDT classifier and Support Vector Machine (SVM)[10], since both of them are widely used. Based on our two-fold cross-validation analysis, we concluded that GBDT

classifier performs better than SVM. The classification results are shown in Table 2. Here are two pinpoints about GBDT classifier.

Table 2: SVM & GBDT classifier comparison on user intent classification accuracy

Class	SVM	GBDT	Class	SVM	GBDT
0	0.8151	0.8271	7	0.6909	0.7268
1	0.7995	0.8314	8	0.8768	0.8777
2	0.5728	0.5851	9	0.8311	0.8943
3	0.9318	0.9467	10	0.7345	0.8718
4	0.7431	0.7789	11	0.8901	0.9492
5	0.9189	0.9321	12	0.7562	0.7623
6	0.8990	0.9536	13	0.3246	0.3199

- The classes are semantically different, but the questions can be very similar in terms of structure. Tf-idf and word2vec[11] features are combined to construct a complex feature space to strike the problem.
- The training set cannot include every possible utterance by users. Hence, generalization of the training data is crucial to avoid classification errors. The classifier has to learn the structure or pattern of requests and not depends on specific keywords. For example, adding training data such as 'What is the rating of X" or 'How to cook Y' can help classifier to learn the structural parts of user requests and eventually perform much better.

4. Entity classifier is added as a second-level classification to improve the overall accuracy of intent classification. Any utterance that does not belong to a specific component is classified as 'Transition' or 'Unrecognized' from top-level GBDT classifier. Most of these requests share similar patterns such as 'Let's talk about X'. The entity 'X' can be one of our supported topics, person, movie title, celebrity, etc. These types of requests are special because the meaning is not based on any structure, but solely based on a certain entity. For example, GBDT classifier cannot classify the request 'Let's talk about Bruno Mars' because it does not know if Bruno Mars is related more closely to the music component or Wikipedia component. These requests are relabeled again by entity classifier and sent back to the corresponding component. Google Knowledge Graph API and cosine distance of words are used as main features to classify different types of entities. The first level and the second level classes are shown in Table 3.

Table 3: User intent classification labels

(a) For the first-level GBDT general classifier

Label	Class	Label	Class
0	Positive	7	LiveQA
1	Negative	8	Movies
2	Small-talk	9	Music
3	News	10	Opinions
4	Wikipedia	11	Food
5	Weather	12	Transition
6	Jokes	13	Unrecognized

(b) For the second-level entity classifier using Google Knowledge Graph API

Label	Type	Label	Type
0	News	8	Hockey
1	Music	9	Animal
2	Movie	10	Science
3	Food	11	Space
4	Basketball	12	Technology
5	Soccer	13	Health
6	Baseball	14	Location
7	Football	15	Celebrity

In the above table, 13 classes of component level classes and 15 classes based on Google Knowledge Graph are listed in the sub-tables. In this way, the intent classification process becomes more accurate.

Currently, the classifier, incorporated with the Google Knowledge Graph API, works well to guide utterances to desired component. Figure 4 and Figure 5 are the ratings and length of conversation over the period from June 15th to August 30th with clear marks where changes are made to the classifier.

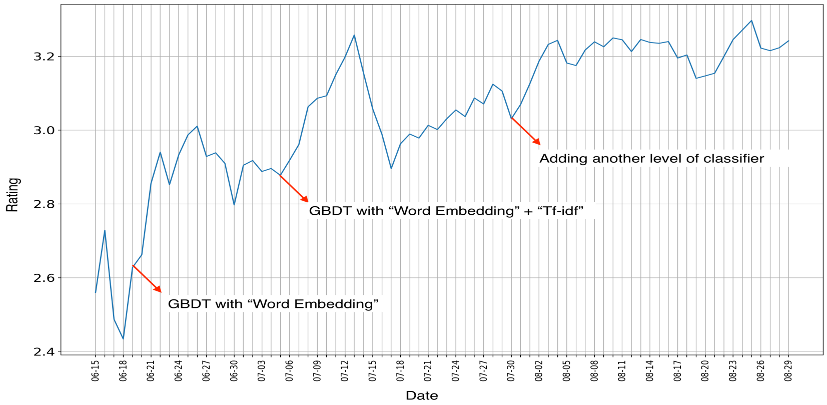

Figure 4: Effect of intent classification on ratings (x-axis: Date, y-axis: Rating).

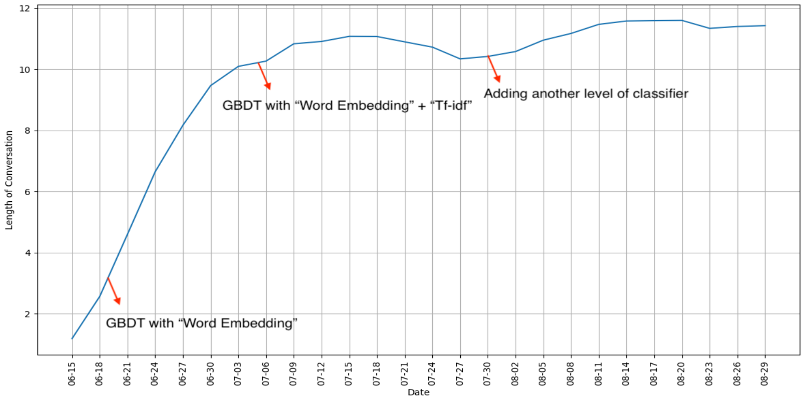

Figure 5: Effect of classifier on length of conversation (x-axis: Date, y-axis: Length of conversation on average in one session).

We sampled the last two days’, around 629 turns of conversations with Alexa users, and only 15 turns were classified incorrectly. So, the accuracy of classification in the real situation is around 97.62% currently. It is clearly shown in the above figures that the ratings are improved instantly when the classifier is improved. The length of conversation tends to be stable while reaching a certain number of turns.

4.2 Effect of proactiveness on performance

When designing our conversational AI, controlling the proactiveness of our bot was a very interesting topic. Will users be more satisfied if the bot is active and drives the conversation, or will users prefer the bot to be passive because they want to be in control of the flow. Based on our four modifications and log analysis, we concluded that user satisfaction is higher when the system behaves proactive than passive. Figure 6 and Figure 7 illustrate how these changes are directly related to user satisfaction. The x axis is Dates, the y axis are Ratings and Length of Conversation, the marks are pinpoints where improvements are made. Rating data and conversation data from June 15th to August 30th are used for analysis.

In the above figures, four pinpoints are pointed out as described below:

- Pinpoint 1 - Passive. A general ending, "what would you like to talk about?" is attached to each response. The bot is very passive because it waits for the user to start a new topic.

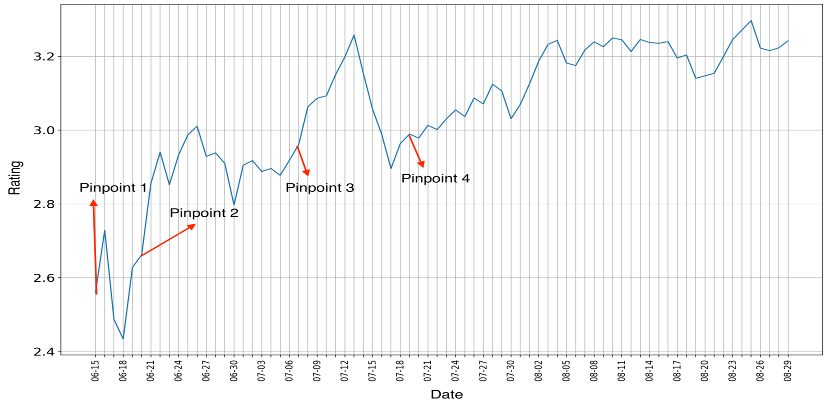

Figure 6: Effect of Proactiveness on rating (x-axis: Date, y-axis: Rating).

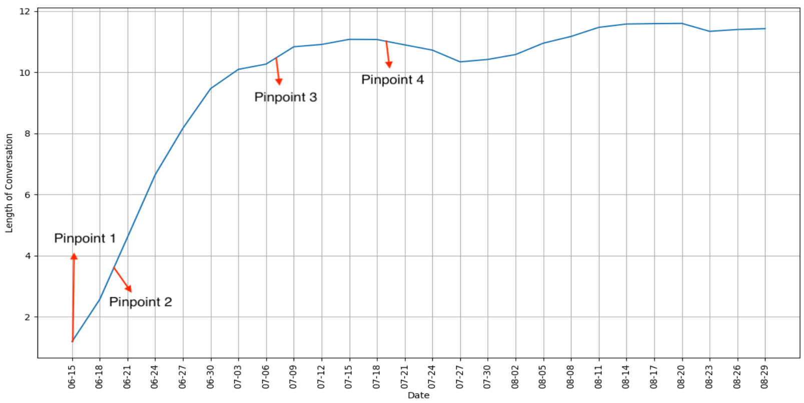

Figure 7: Effect of Proactiveness on length of conversation (x-axis: Date, y-axis: Length of conversation on average in one session).

- Pinpoint 2 - Less Passive. "What would you like to talk about, movies, music, news, or food?" is attached to each response. It became less passive since the bot starts to direct users to talk about specific topics. However, it became repetitive after several turns of conversation and decreased user satisfaction.
- Pinpoint 3 - Less Active. One specific component related topic randomly suggested to users after each turn of conversation makes the conversation more natural. Whenever user is finished with current topic, the bot tries to direct users to talk about a different topic. However, choosing new topics does not incorporate previous conversation and we do not have an information about if user will enjoy this topic or not. Therefore, on average, the satisfaction score increased but some users still gave low ratings.
- Pinpoint 4 - Active. Expanded the suggestion features to many different topics such as sports, technology, science and celebrity. Based on the current conversation, the bot will suggest headline news about those topics, short fun facts about science or animal, news about outer space, celebrity gossip, and more. Since the range of supported topics increased, the satisfaction score increased further.

4.3 Effect of engagement within a component on performance

Whether each component has the ability to engage with users for several turns makes a difference on users' experience. For example, Figure 8 shows the length of conversation within each component during each conversation on average. For music, movies, and food, as the ability of conducting continuous conversation is improving, the number of turns about these topics increases correspondingly. Since the news component was ready way before other components, the number of turn was much larger at the beginning. As other components were developed gradually, the time spent on each topic was assigned more evenly. In the end, the number of turns on news, movies, food, and music converges. Conversation data from June 15rd to August 30th are used for analysis.

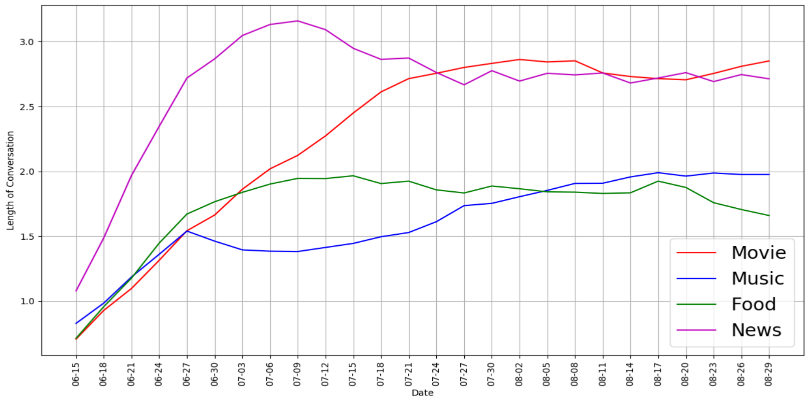

Figure 8: Length of conversation within each component (x-axis: Date, y-axis: Length of conversation within each component on average in one session)

The above analysis has weakness of independence. While significant changes are made to classifiers and proactiveness, other changes are made to components as well. Since the competition was alive, it was not feasible to do A/B testing. The argument would rather be that changes made to specific components overall are gradual, and up and downs are reflections of the process of perfecting the conversation logic and refining each component. So, significant changes made to the classifier and the proactiveness is legitimately reflected on the sudden and consistent improvement of the conversation performance shown in aforementioned graphs.

5 Conclusions and future work

The main contribution is a combination of state-of-the-art question answering and information retrieval components within an open-domain dialogue manager, designed to maintain the flow of conversation. The conversation logic in the dialogue manager is designated to cope with a wide range of situations in the conversation to make sure that the conversation flow is maintained.

In our analysis, we found that the performance of a conversational AI is crucially affected by a number of factors, such as automatic speech recognition, natural language processing, intent classifier, response patterns, interaction strategies, etc. In our detailed analysis, three significant factors are revealed that impact the agent's performance significantly. Intent classifier is critical for an information retrieval oriented system, since different components are designed to deal with different questions, and incorrect intent classification may cause incoherent interactions with users. A classifier with high accuracy gives users more satisfied experience. The proactivity mechanism is effective to lead the conversation dynamically by recommending topics that are related to the current conversation. The capacity of engagement on a specific topic draws users closer to a discussion, which reflects the intelligence of the agent.

In summary, Emersonbot is a valuable step towards open-domain conversational AI, focusing on conversational search, by integrating components involving different areas of expertise. Our system performed competitively in the Alexa Prize 2017 Challenge, ending with a rank of 5th among all teams.

Our current and future work will focus on improving the ability of engaging users in conversations, including the capacity of engagement, the proactivity mechanism, and the expansion of supported topics.

References

[1] Shawar, B. A. A., & Atwell, E. (2005). A Corpus Based Approach to Generalising a Chatbot System (Doctoral dissertation, University of Leeds).

[2] Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., & Jurafsky, D. (2016). Deep reinforcement learning for dialogue generation. arXiv preprint arXiv:1606.01541.

[3] Ji, Z., Lu, Z., & Li, H. (2014). An information retrieval approach to short text conversation. arXiv preprint arXiv:1408.6988.

[4] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014, June). The stanford corenlp natural language processing toolkit. In ACL (System Demonstrations) (pp. 55-60).

[5] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. Text Mining: Applications and Theory, 1-20.

[6] Fuzzy String Matching in Python. (2011, July 8). Retrieved from http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python.html

[7] Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., & Jurafsky, D. (2013). Deterministic coreference resolution based on entity-centric, precision-ranked rules. Computational Linguistics, 39(4), 885-916. Chicago

[8] Artificial Intelligence Markup Language (AIML). (2011, October 30). Retrieved from http://www.alicebot.org/TR/2011.html

[9] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

[10] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

[11] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[12] Heisele, B., Serre, T., Prentice, S., & Poggio, T. (2003). Hierarchical classification and feature reduction for fast face detection with support vector machines. Pattern Recognition, 36(9), 2007-2017. Chicago