Chatbot Evaluation and Database Expansion via Crowdsourcing

Zhou Yu, Ziyu Xu, Alan W Black, Alexander I. Rudnicky
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, 15217
{zhouyu, air, awb}@cs.cmu.edu, ziyux@andrew.cmu.edu

Abstract

Chatbots use a database of responses often culled from a corpus of text generated for a different purpose, for example film scripts or interviews. One consequence of this approach is a mismatch between the data and the inputs generated by participants. We describe an approach that while starting from an existing corpus (of interviews) makes use of crowdsourced data to augment the response database, focusing on responses that people judge as inappropriate. The long term goal is to create a data set of more appropriate chat responses; the short term consequence appears to be the identification and replacement of particularly inappropriate responses. We found the version with the expanded database was rated significantly better in terms of the response level appropriateness and the overall ability to engage users. We also describe strategies we developed that target certain breakdowns discovered during data collection. Both the source code of the chatbot, TickTock, and the data collected are publicly available.

Keywords: crowdsourcing, dialog systems, appropriateness

1. Introduction

Chatbots have recently become the focus of greater research interest. Unlike goal oriented dialog systems, chatbots do not have any specific goal that guides the interaction. Consequently, traditional evaluation metrics, such as task completion rate, are no longer appropriate. The difficulty of evaluation is intrinsic as each conversation is interactive, and the same conversation will not occur more than once; one slightly different answer will lead to a completely different conversation; moreover there is no clear sense of when such a conversation is “complete”. It is not possible to design a pipeline to evaluate such systems in a batch mode, nor is it easy to equate participants on various dimensions that may influence their behavior.

In addition to the difficulty of evaluating a chatbot, another challenge is identifying an appropriate database. Ideally, it should be a corpus that has the same distribution as the future users’ conversations. However, if we are not designing a specific system for a targeted group, but rather a system that can be used by a variety of different users, the best strategy for designing a response database is not clear.

We describe an approach we have developed to the problem of creating a database of useful responses that makes use of an existing corpus as a base but also incorporates a process that iteratively transforms the database into a form that is better suited to the chat domain. We do this by crowdsourcing the appropriateness of responses (in given conversations) and by soliciting improved responses.

2. Related Work

Current chatbots use a variety of methods to generate responses, such as machine translation (Ritter et al., 2011), retrieval based response selection (Banchs and Li, 2012), and recurrent neural network sequence generation (Vinyals and Le, 2015). Yet, the databases they use to power their systems have very little variability. Some systems used micro-blogs, such as Twitter conversations (Ritter et al., 2011) and some used movie subtitles (Banchs and Kim, 2014; Ameixa et al., 2014; Banchs and Li, 2012), and there is research that used Twitter as a database but switched to ask the human to generate responses in the crowdsourcing platform in real time when the database failed to have an appropriate response (Bessho et al., 2012). Most of the work reported above have no real user evaluation or a small group of people for evaluation. Only two kinds of databases have been used, movie subtitles and micro-blogs. In this work, we focus on how to generate appropriate databases for chatbots and conduct evaluations for chatbots by leveraging crowdsourcing resources.

3. TickTock System Description

TickTock is a system that is capable of conducting free-form conversations, in contrast to goal-driven systems, which are designed to acquire information, provide feedback, or negotiate constraints with the human. A free-conversation system in principle removes any built-in value for the human and its success depends on the machine keeping the human interested in the ongoing conversation. Thus, as task completion is no longer an applicable metric, we chose to focus on metrics of the user’s experience, such as engagement, likability, and willingness to future interaction. TickTock is an IR-based system with conversation strategy facilitation. A multimodal version of TickTock is described in (Yu et al., 2015), with similar architecture but with minor adjustments to the conversational strategies.

TickTock 1.0 has a database that consists of question-answer pairs from CNN Interview Transcripts from the “Piers Morgan Tonight” show. The corpus has 767 Interviews in total and each interview is between 500 to 1,000 sentences. To construct our database, we used a rule-based question identification method, which simply means searching for tokens such as ‘?’, ‘How’, ‘Wh-’, etc. to identify questions and then extracted the consecutive utterance of the other speaker as the answer to that question. In total we have 67,834 pairs of utterances.

Key Term Matching (Martin, 2002) was used for response generation. The user’s text input is first processed by the NLU component, which performs POS tagging (Toutanova et al., 2003) and removes stop words; heuristics are then used to compute the database and calculate the weighted sum, which becomes the retrieval confidence score. Finally, we normalize the score by dividing it by the length of the retrieved utterance. We filter out inappropriate content, excluding the retrieved answer if it is longer than 15 words and remove other characters such as parentheses or square brackets (along with everything between them). Our goal is to generate coherent conversations without deep understanding of the context, which is useful in a non-task oriented interactive system, and is motivated by lexical cohesion in modeling discourse. The coherence can be reflected by the repetition of lexicon items. The method first does shallow syntactic analysis of the input utterance and extracts keywords. These are used to search the corpus for a suitable response. Once we retrieved the response, we select a conversational strategy, based on a heuristic, i.e. a predefined threshold for the retrieval confidence score, which can be tuned to make the system appear more active or more passive.

Higher thresholds correspond to more active user engagement. When the retrieval confidence score is high, we return the found response in the database back to the user. If the retrieval confidence score is low, meaning no good response was obtained, we use strategies to change the current topic by randomly choosing four types of conversation strategies we designed. "Proposing a new topic", such as "sports" or "music"; "Closing the current topic using an open question, such as "Could you tell me something interesting?"; "Telling a joke", such as " Politicians and diapers have one thing in common. They should both be changed regularly, and for the same reason"; and finally "Initiate things to do together", such as " Do you want to play a game together?"

4. Methodology

The purpose of the crowdsourcing study is twofold. The first is to collect diverse conversations from a large number of people. The second is to expand TickTock's database, so it has more targeted question-response pairs. The TickTock system is implemented in Python, making it platform independent. We adapted TickTock to a web version through a web socket connection to a web page implemented in PHP. People can get access to TickTock through any browser. We made the source code of TickTock, a web-based demo and a collection of 100 conversations on Mechanical Turk publicly available here[1].

We designed three crowdsourcing tasks to expand TickTock's database. The first task is "the conversation generation task", in which a user interacts with TickTock by typing. The second task is "the conversation rating task", in which the user rates how appropriate TickTock's response is per conversational turn. The third task is "the conversation correction task", in which the user generates appropriate responses for TickTock. For the last task, we only selected the conversational turns that were rated not appropriate in the second task for correction by Turkers.

We recruited participants on the Amazon Mechanical Turk Platform with Turker criteria of: higher than 95% life time approval rate, completed more than 50 hits, and located in the United States.

After we had collected sufficient data from the above three tasks, we expanded our database by adding the human approved high-appropriateness responses obtained from the second task and the human corrected responses obtained from the third task to TickTock's database. The system with the expanded database is named TickTock 2.0. The new version was then put on the Amazon Mechanical Turk Platform to collect more data. After we collected more conversations and the corresponding appropriateness ratings for those conversations, we calculated the appropriateness rating distribution of the two versions of TickTock to determine if there was improvement after we expanded the database with more targeted data.

We also collected user subjective ratings for the two versions. We asked users how engaged they felt over all in their interactions. Users who interacted with both versions of the system were also asked whether they preferred TickTock 2.0.

5. Mechanical Turk Study Designs

We designed three tasks and recruited participants on the Amazon Mechanical Turk Platform. We describe the task design and the collected data in this section.

5.1. Conversation Generation Task

We asked the user to interact with TickTock via the web page for at least 10 conversational turns. We also collected information from users on whether they had interacted with a chatbot before. Users were also asked to state what they liked, and disliked about the task; this was for purposes of providing insight for system improvement. The task starts when user types in an utterance on a web page, and submit it to the server, the server then fetches a response and displays it to the user. For both versions, we collected 50 conversations. For TickTock 1.0, this took over four days, with 44 participating Turkers (we allow people to do the same task multiple times), resulting in 589 conversation turns. The conversation length on average was 11.9, with a standard deviation of 1.9. With TickTock 2.0, it took over ten days, with 29 Turkers, and resulted in 590 turns. The conversation length on average was 11.8, with a standard deviation of 4.2. We are not certain why the two versions took different amounts of time; perhaps our task was of limited overall interest.

5.2. Conversation Rating Task

Once the Turker finished his or her conversation with TickTock, we asked them to rate how appropriate they felt the system's responses were with respect to their inputs. We also told them to make the decision for each conversational turn. Table 1 describes the annotation scheme that Turkers were given. The Turker would see the entire dialog on the web page, and were asked to choose between three labels: 'Inappropriate', 'Interpretable' and 'Appropriate'.

We randomly sampled 10 percent of the collected utterance pairs and asked an expert to rate how appropriate TickTock's responses are given the same coding manual Table 1. Since we wanted to collect conversational turns

[1] http://www.cs.cmu.edu/afs/cs/user/zhouyu/www/TickTock.html

Label	Definition	Example
Inappropriate	Not coherent with the user utterance	*Participant*: How old are you? *TickTock*: Apple.
Interpretable	Related and can be interpreted	*Participant*: How old are you? *TickTock*: That's too big a question for me to answer.
Appropriate	Coherent with the user utterance	*Participant*: How is the weather today? *TickTock*: Very good.

Table 1: Appropriateness rating scheme.

that are not appropriate and send them back to Turkers to generate more appropriate responses for TickTock, we collapsed 'Inappropriate' and 'Interpretable' into 'Not Appropriate' when doing the annotation agreement as we wanted to distinguish between 'Not Appropriate' and 'Appropriate'. The agreement of the participant's self-rated appropriateness and the expert-rated appropriateness has a kappa of 0.73. In Table 2, we display an example dialog with its associated appropriateness ratings.

5.3. Conversation Correction Task

Turkers were shown three utterances: one utterance from the participant, one utterance from TickTock and another utterance from the participant. Then, they were asked to type in what they should say if they were TickTock given the three utterances. The original TickTock's response to the previous utterance from the participant was not shown. In total, 28 Turkers participated in this task.

We randomly sampled 10 percent of the corrected conversational turns and asked an expert to rate how appropriate the responses were, according to the appropriateness rating scheme mentioned above. We found that 82.8% of the responses were appropriate, and the inappropriate responses were just answers with different lengths of the letter 'd', which is what people put to get through the task in the most efficient way. We filtered these bad responses out based on a simple regular expression. The appropriate percentage thus increased to 100%. We conjecture that the reason this task appears very easy for Turkers might be their experience in conducting conversations with others.

6. Results and Analysis

We would judge that our approach is a reasonable way to generate focused chat data: We spent approximately $50 for the experiments we conducted on the Amazon Mechanical Turk Platform and collected 50 conversations for each of the two versions of the system. The experiments lasted 14 days in total. As one would expect, we found it usually takes less time to complete the task if you raise the payment of the task and is therefore a decision that researchers can make according to their priorities. In Table 3, we show the distribution of the rating of two systems' response appropriateness. The inappropriateness ratio of turns has dropped from 55% to 34% by expanding the database with appropriate question-answer pairs.

There are in total eight people who have interacted with both versions of TickTock, in which five of them preferred the second version, two of them think the two versions are the same, and the remaining one was not sure on their preference. The average user self-reported engagement score is 2.4 (out of 5) in the TickTock 1.0 experiment and 3.6 (out of 5) in TickTock 2.0. We observe that TickTock 2.0 is performing better than TickTock version 1.0 in both per turn appropriateness measure, and per interaction user engagement measure. There are only around 10 percent of users (8/(44+29)) who interacted with both versions of the system. In the future, we would like to assemble a pool of Turkers that we can continue asking to interact with different versions of the chatbot, for evaluation consistency.

After filtering stop words, we did a word count of all the utterances users typed in and found that none of the top 50 ranked words were content words. Most of them were functional words, such as 'you', 'I', 'is', 'do', 'are', 'what', similar to what has been observed in human conversation.

We also performed a POS tagging and extracted the nouns used by participants; we found the top ranked words to be 'music', 'movie', 'game', 'sense', 'name', 'TickTock', and 'Trump'. We configured TickTock to initiate conversation about everyday topics such as music and movies when lacking an appropriate answer, which may have led participants to expand on those topics in conversation. Some participants got annoyed and said 'You don't make sense' or 'It doesn't make sense', causing "sense" to be at the top of the list as well. We calculated the percentage of pronouns used in user sentences. The percentage, 9.4%, is relatively high. This is another phenomenon that seems to be related to the word choice in casual conversation.

We also went through a randomly selected 10% of the data, and observed that the quality of the users' input is not always very satisfying. Most of the users failed to provide concrete inputs to the conversation. They asked Ticktock simple questions or simply went along with whatever TickTock was saying instead of pushing the conversation topic deeper. How to elicit high quality user input is a difficult question; very few researchers in the chatbot community appear to have paid attention to it. We would argue that not only is an appropriateness metric for TickTock's response needed, but also an additional metric that evaluates the quality of a user's input. Together they may better characterize chatbot performance.

7. Strategies Designed for Different Types of Breakdowns

We randomly sampled 10% of the conversations generated by using TickTock 1.0 and classified the nature of conversational breakdowns (turns given ratings that are not 'Appro-

Participant	TickTock(Appropriateness)
Hello how are you?	I'm feeling quite good. (3)
Anything new going on with you?	Of course. What do you think? (3)
I am not sure. What's on your mind?	Well, you know, I'm not sure. let me tell you something, Politicians and diapers have one thing in common. They should both be changed regularly, and for the same reason. (3)
Wow. That funny and insightful.	Uhh, I can't answer this. Can we talk about something else? (2)
How about movies. Have you seen any good movies?	Of course. What do you think? (2)
What have you seen?	Yes. (2)
I love super hero movies. Have you seen Deadpool?	Uh-huh, I do. (2)

Table 2: An example conversation with TickTock

Version	Inappropriate	Interpretable	Appropriate
1	321 (55%)	138 (23%)	130 (22%)
2	200 (34%)	242 (41%)	148 (25%)

Table 3: The distribution of appropriateness ratings of two versions of TickTock.

priate') into five types. We formulated targeted strategies for each type and evaluated them on the data collected by TickTock 2.0.

1. **Single-word Sentence**: We found that some users were typing in meaningless single words such as 'd', 'dd', or equations such as '1+2='. TickTock will reply 'Can you be serious and say things in a complete sentence?'. We have a set of surface realization of such replies to choose from, so users would get a lightly different version every time, with the aim of making TickTock seem less robotic. It triggered 12 times in the TickTock 2.0 generated conversations.

2. **Out of Vocabulary**: We found that typos occur in the users' responses and they used words that are not in the vocabulary of our database, such as 'confrontational'. We implemented a strategy that when a sentence contains an out of vocabulary word, TickTock will reply with a clarification question, such as 'What is 'confrontational'?' to communicate that it cannot understand his utterance entirely. It triggered 36 times in the TickTock 2.0 generated conversations.

3. **Anaphora**: We found user inputs with very limited concrete information in themselves, but referred to a prior response in the conversation. An example input would be "I hate them" and it is referring back to the 'sports' topic in the previous phrase, "How about we talk about sports?". Anaphora is a difficult problem to solve for complex sentence structures. However in colloquial sentences, substituting in the noun of the previous sentence covers 85% of the cases. We implemented this simple rule to tackle anaphora. It triggered 30 times in the TickTock 2.0 generated conversations.

4. **Query Knowledge Base for Named Entities** A lot of Turkers assumed TickTock could answer factual questions, so they asked questions such as "Which state is Chicago in?". We used the Wikipedia knowledge base API to answer such questions. We first performed a shallow parsing to find the named entity in the sentence, which we then searched for in the knowledge base, and retrieved the corresponding short description of that named entity. We then designed a template to generate sentences using the obtained short description of the mentioned name entity, such as "Are you talking about the city in Illinois?". It triggered 22 times in the TickTock 2.0 generated conversations.

5. **Weight adjustment with tf-idf** We re-weighted the importance of the key words in an utterance based on its tf-idf score. Using POS tagging of the words that match between a user input, and the sentence a response is in reply to, we give nouns a score of 3, verbs a score of 2, and other words a score of 1. We then multiply each of these scores by the tf-idf value of the corresponding words, and the sum of their scores gives us the score of the response.

8. Conclusions and Future Work

We found that using suitable designed crowdsourcing tasks, we can expand TickTock's database with more targeted response pairs. The version using the expanded database was preferred by most of the users and was better rated in terms of response appropriateness and the overall ability to engage users. We also found it is feasible to use the crowdsourcing platform for system evaluation. An analysis of the data we obtained also allowed us to define strategies to recover from breakdowns (some of which have previously been reported by others).

Our intent is to go beyond the response appropriateness and put more emphasis on overall discourse cohesion. For example, there is a breakdown type we have not addressed, which is the chatbot's inconsistency in adhering to the context of the conversation. A possible solution would be to maintain a knowledge base of what the user said and use it for consistency checking as part of the selection process for the final response.

We are also interested in determining how the system can channel a conversation into a specific topic. That is, if TickTock starts the conversation with a given topic, how long and with what strategies will it be able to keep the user on the same topic. We also wish to develop strategies that elicit high quality responses from human users (perhaps as a consequence of maintaining a high level of engagement).

9. Acknowledgements

We thank Peter Xiang Gao for technical supports and discussions.

10. References

Ameixa, D., Coheur, L., Fialho, P., and Quaresma, P. (2014). Luke, i am your father: dealing with out-of-domain requests by using movies subtitles. In *Intelligent Virtual Agents*, pages 13–21. Springer.

Banchs, R. E. and Kim, S. (2014). An empirical evaluation of an ir-based strategy for chat-oriented dialogue systems. In *Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)*, pages 1–4. IEEE.

Banchs, R. E. and Li, H. (2012). Iris: a chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL 2012 System Demonstrations*, pages 37–42. Association for Computational Linguistics.

Bessho, F., Harada, T., and Kuniyoshi, Y. (2012). Dialog system using real-time crowdsourcing and twitter large-scale corpus. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 227–231. Association for Computational Linguistics.

Martin, J. R. (2002). *Meaning beyond the clause: area: self-perspectives*. Annual Review of Applied Linguistics 22.

Ritter, A., Cherry, C., and Dolan, W. B. (2011). Data-driven response generation in social media. In *Proceedings of the conference on empirical methods in natural language processing*, pages 583–593. Association for Computational Linguistics.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.

Vinyals, O. and Le, Q. (2015). A neural conversational model. ICML Deep Learning Workshop 2015.

Yu, Z., Papangelis, A., and Rudnicky, A. (2015). Tick-Tock: A non-goal-oriented multimodal dialog system with engagement awareness. In *Proceedings of the AAAI Spring Symposium*.