TickTock
RE-WOCHAT 2016 Shared Task Chatbot Description Report

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

This is a description of the TickTock chatbot system, which is a retrieval based system that utilizes conversational strategies to improve the system performance. It has two versions, one with multimodal signals as input; one with text input through typing. The multimodal version is a stand alone system (Yu et al., 2015), while the text version is a web-API version. In this report, we focus on describing the web-API version of TickTock, which is used in the shared task.

1. General 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 along with the appropriateness of the system responses. Similar to (Banchs and Li, 2012), TickTock is a retrieval based non-goal oriented dialog system. It generates the response by utilizing the corpus, which is a interview corpus. Different from other retrieval systems, it has strategies that handle situations where the retrieval methods could not produce appropriate responses.

2. TickTock System Description

TickTock is an retrieval 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 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. Later we recruited users in Mechanical Turk to generate targeted utterances for this specific task (Yu et al., 2016).

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 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. We used six strategies described below to address some of the issues within the user input.

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.

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.

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.

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 1: An example conversation with TickTock

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?".

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.

6. **Incorporating One-utterance History** In ranking the retrieved response, we Incorporated the previous one turn context of the conversation. We compute the cosine similarity of the highly ranked response with the previous utterance, and picked the one that is more similar. We convert the utterances to vector space using word2vec method.

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, the dialog manager will choose a strategy that takes context into consideration. There are five strategies we used to deal with possible breakdowns the low retrieval confidence score indicates:

1. **Switch topics**: propose a new topic other than the current topic, such as "sports" or "music".

2. **Initiate things to do**: propose something to do together, such as "Do you want to see the latest star war movie together?".

3. **End topics with an open question**: close the current topic using an open question, such as "Could you tell me something interesting?".

4. **Tell a joke**: tell a joke such as: " Politicians and diapers have one thing in common. They should both be changed regularly, and for the same reason".

5. **Elicit more information**: ask the user to say more about the current topic, using utterances such as " Could we talk more about that?â.

3. An Example Conversation

In Table 1, we show an example conversation that TickTock produced.

4. Future Work

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).

5. References

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.

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

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

Yu, Z., Xu, Z., Black, A., and Rudnicky, A. (2016). Chatbot evaluation and database expansion via crowdsourcing. In *Proceedings of the chatbot workshop of LREC*.