Response Generation for Multi-Party Conversations

Stanford CS224N Custom Project

Isaac Bevers
Department of Computer Science
Stanford University
ibevers@stanford.edu

Matt King
Department of Computer Science
Stanford University
mcking@stanford.edu

Abstract

In the past several years, chatbots for one-on-one conversations have seen great success. However, chatbots for multi-party conversations are under-studied. We explore whether substantial performance gains on multi-party chat tasks can be achieved by by fine-tuning. Specifically, we compare the performance of basic BlenderBot, a state-of-the-art chatbot designed for two-party conversation, to BlenderBot fine-tuned on multi-party chats from the Ubuntu Dialog Corpus.

1 Key Information

- Mentor: Ethan Chi
- External Collaborators: None.
- Sharing project: No.

2 Introduction

In 2001, 'Smarterchild,' an entertaining chatbot with access to various external data sources was released on AOL Instant Messenger and MSN Messenger. Since then, the prevalence of chatbots has increased greatly. We now regularly use intelligent personal assistants such as Siri, Alexa, and Google Now. Chatbots are regularly found in customer-support systems and have been integrated into other messaging platforms such as Facebook Messenger and Slack [1].

However, current research on chatbots focuses almost exclusively on chatbots that interact with one other (human) conversant at a time. This gap between chatbot performance in the two-party setting and the multi-party setting is not for lack of compelling applications of chatbots in multi-party settings (discussed below) but because it is harder to learn generalized representations of multi-party dialogue and because are more poses a more difficult problem. Indeed, at the 2019 NeurIPS conference, the "Eighth Dialogue System Technology Challenge" recognized chatbots for multi-party conversations as being of "broad interest" with "practical impact" and at the forefront of research in the field[2].

Oe large, recent step forward was given by Wang et. al. in 2020: the paper "Response selection for multi-party conversations with dynamic topic tracking" both reframes the multi-party chatbot problem as having two subtasks (topic disentanglement and response selection) and provides a novel model architecture which outperforms existing multi-party chatbots in several automated metrics [3]. Here, we make fewer drastic changes to the two-party chatbot problem. In fact, we deliberately frame our project so as to evaluate the effectiveness of a single innovation (implemented from scratch) for the multi-party setting. This innovation is the conceptualization of any chatbot configured for two-party conversation as a (potentially poor) chatbot capable of participating in genuinely multi-party conversation. This paradigm and the associated chatbot architecture immediately yield a natural, general method of evaluating how effective a two-party conversation chatbot is at multi-party conversation. This informs our baseline, Facebook's BlenderBot as trained on two-party data. Further,

as our primary contribution, we use the paradigm of a two-party chatbot as a multi-party chatbot, to finetune BlenderBot on multi-party data, specifically the Ubuntu Dialog Corpus.

3 Related Work

Up until recently, most chatbot research was focused on two-person conversation. In their comprehensive review of chatbot research in 2019, Steering et al. found that approximately 90% of chatbot papers found using the keyword 'bot' were about two-agent bots, while only 2.5%, or 104 were about multiparty chatbots.

To understand the possible roles that multiparty chatbots can fill, Steering et al. used an ideation method to generate about 600 ideas for multiparty chatbots. Then they used Affinity Diagramming to categorize these chatbot ideas, resulting in following categorizations: Organizer, Host, Adviser, Archivist, Assistant, Antagonist, Novice, and Regular user. They observe that many chatbots are designed to do a highly specific task. However, it's likely that there are representations that are common to all of them, suggesting that a pre-trained model fine-tuned on task-specific data would likely perform better than one trained solely on task-specific data [4].

One way to improve multi-party chatbots is to improve the models ability to select a good response by disentangling sub-conversations in a multi-party chat. Performance has consistently improved on these tasks. In 2020, Wang et al. created a BERT-based dynamic tracking model that selected the correct response in examples from the DSTC-8 Ubuntu dialogue corpus 72.6 percent of the time and correctly disentangled conversational sub-topics 75.4 percent of the time, setting new benchmarks for these tasks [3][5]. While these tasks may be key to maximizing the performance of multi-party chatbots, they cannot be successful unless they are combined with a chatbot that generates good responses in the first place. Accordingly, we see our approach as complementary to Wang et al.'s.

4 Approach

Our approach is motivated by the desire to leverage extensive literature on chatbots for two-party conversations (one chatbot and one human) and apply this to the nascent field of chatbots for multi-party conversations. We frame the problem of creating chatbots for multi-party conversation more precisely as follows: can we teach state-of-the-art two-party-chatbots to converse in the multi-party setting via fine-tuning on multi-party chat data? This approach both makes clear how we plan to approach the problem of multi-party chatbots and it suggests a well-motivated baseline. Our baseline model will use Facebook's highly successful BlenderBot [6], designed for two-party conversation.

Data for training and evaluating two-party or multi-party chatbots comes in the form of **episodes**, formed from a conversation between humans. In a single episode each participant may speak one or many times. Hereafter, the term episode has the technical meaning of a complete exchange between conversants which will yield at least one training example, whereas the word conversation is used informally. Intuitively, to evaluate the performance in the multi-party setting of a chatbot which has been trained to generate responses in the two-party setting, one needs a universal way of representing a multi-party conversation as a two-party conversation. This connects to the question of how to train or evaluate on an episode. For each conversant in an episode who does not speak first, we create a labeled example from the episode by replacing that conversant with the chatbot. Figure 1 below captures this process.

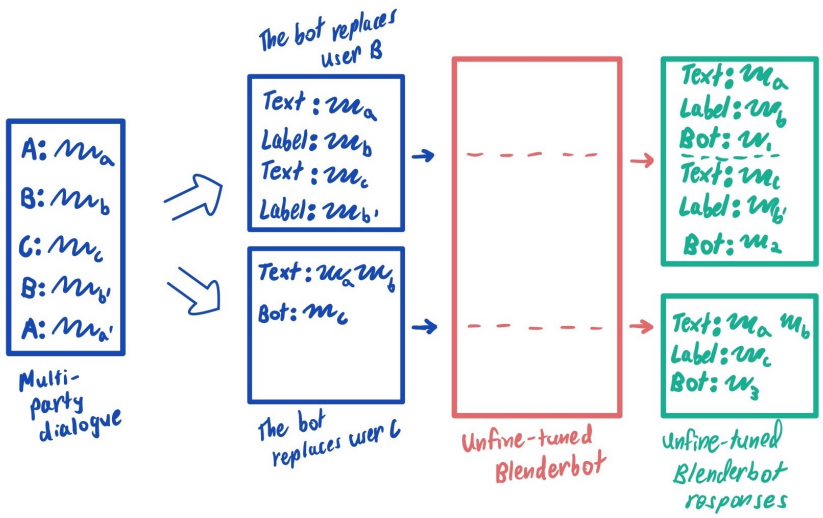

Figure 1: **Baseline BlenderBot Evaluation Pipeline.** The left column features an un-modified multi-party conversation. In the second column, 2 episodes are created. In the first episode, the chatbot 'replaces' conversant B, who speaks twice, giving 2 labeled examples in this episode; In the second episode, the chatbot 'replaces' conversant C, who speaks once, giving 1 labeled example. The text field of this labeled example exhibits the discussed concatenation of utterances from conversants A and B.

For our main experiment, we fine-tune BlenderBot on the Ubuntu Dialogue Corpus. Here, there is a fundamental difference in the input format of data. When evaluating the baseline model, which had no previous exposure to multi-party data, there is no performance to be gained by including which non-chatbot conversant said what. By contrast, intending that a fine-tuned BlenderBot ultimately learns to use the information of which non-chatbot conversant is speaking, we include this in the fine-tuning data, in a format captured in Figure 2 below.

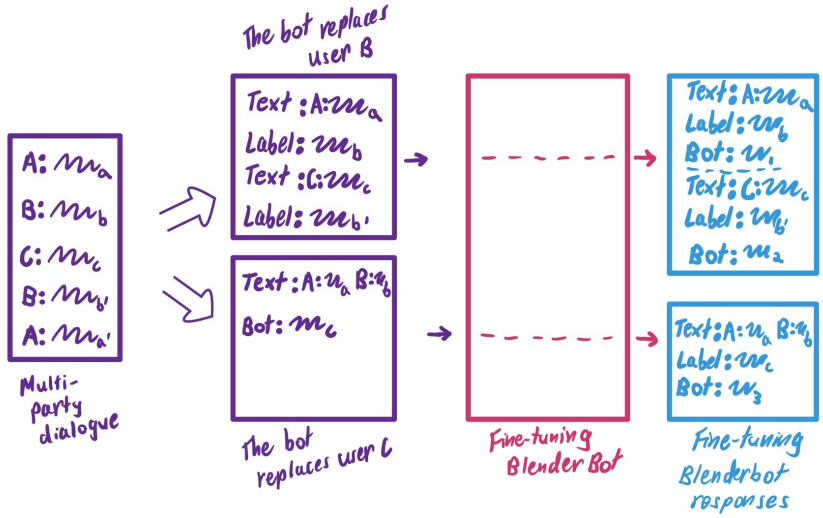

Figure 2: **BlenderBot Fine-Tuning and Evaluation Pipeline.** Compare with Figure 1. The left column features the same un-modified multi-party conversation. All important differences with Figure 1 are in the second column. When the chatbot 'replaces' conversant B, the only change to Figure 1 is that the utterances of conversants A and C are labeled by their speaker. In the second episode, when the chatbot 'replaces' conversant C, the text field does not merely concatenate the two non-chatbot utterances, but delineates which utterance belongs to each speaker.

5 Experiments

5.1 Data

For our dataset, we begin with the same DSTC-8 Ubuntu IRC dataset used by Wang et. al.[7] This dataset includes many conversations about technical topics, but also much idle "chit-chat". Many conversations in the dataset are multi-party (with 3 or more conversants) and we throw out all conversations with 2 or less conversants when preparing our data, forming a dataset of 190000 episodes. This large quantity of examples makes this dataset appealing and even standard for the multi-party chatbot research, and it is the same dataset used by Wang et. al. From the 190000 episodes, we follow the procedure in the first two columns of Figure 1 to form labeled examples within episodes, which yields 290000 labeled examples. Following the format of Facebook's ParlAI python framework, which is compatible with BlenderBot, we transform each Ubuntu chat log labeled examples into one line of the form **text:[non-chatbot speech] labels: [chatbot speech]**.

5.2 Evaluation method

To evaluate our model, it is not sufficient to report only metrics such as perplexity or the conceptually similar metric of mean reciprocal rank (MRR). For a response generator $\mathcal{RG}$ and set E of labeled examples (u, r), we state for ease of reference the standard definition of **perplexity** of $\mathcal{RG}$ on E:

$$Perplexity(\mathcal{RG}, E) = \left(\prod_{(u,r) \in E} \mathbb{P}[\mathcal{RG}(u) = r] \right)^{-1/|E|}.$$

a model with high perplexity, judged to be worse than one with low perplexity, is therefore one unlikely to have produced the label response. Notably, this is essentially independent of any external measure of the "quality" of the model's actual response $RG(u)$ to u. This is simply because external evaluations of the "quality" of responses are notoriously difficult to construct in the response generation setting, when the space of possible responses is so vast.

ACUTE-eval operates on the principle that the average of human opinions of a chatbot's response is a good indication of response quality and provides an important supplement to automated metrics such as perplexity.

To compare the first tuned model to the untuned model, we generate single responses to 20 dialogues from the Ubuntu chat logs for each model. We train human annotators to compare the two models' responses to a given conversation and choose the more relevant, engaging one as a baseline metric of response quality. (The human annotators cannot see which model produced which response). After this blind human evaluation step, we observe which model's responses were selected more often.

5.3 Experimental details

BlenderBot assumes a two-person conversation context, where BlenderBot always responds to it's conversation partners previous utterance. Accordingly, for our baseline we considered multiple approaches to approximating multi-party conversation with BlenderBot. However, the only one viable approach was feeding the entire conversational context to BlenderBot as if it were one utterance. For both the basic version of BlenderBot and the fine-tuned one, we used a 90 million parameter version of BlenderBot.

After generating responses for both BlenderBots on the same set of input dialogues, an annotator (who could not see which response came from which bot) selected the better response for each input.

5.4 Results

Metric	Baseline BlenderBot	Fine-tuned BlenderBot
Perplexity	238.86	9.26
Engagingness*	0.37	0.63
Interestingness*	0.53	0.47
Knowledge*	0.50	0.50
Humanness*	0.53	0.47
Relevance*	0.27	0.73

Here we use the same evaluation metrics suggested by Li et al. in the original ACUTE-eval paper [8]. We add one additional metric, relevance, which we feel is reasonable to include when two of the evaluation criteria - engagingness and interestingness - might otherwise reward a chatbot which gives bizarre, diverting non sequiturs. The one-tailed paired t-test had p-value $p = 0.08$ in favor of the tuned chatbot. We interpret this a reserved indication of success. For many examples we observed the fine-tuned chatbot understand aspects of the multi-party conversation that the unfine-tuned chatbot did not. However, given a (smaller) but significant number of examples on which the fine-tuned chatbot performed worse, and given that statistical rigor demands a two-sided p-test, we do not strongly emphasize the statistical significance.

One example of the fine-tuned chatbot responding a way that suggests it has learned not to view a multi-party conversation as merely two-party is given in Figures 3 and 4. To generate examples for ACUTE-eval we used publicly available compilations of famous literary dialogue. In this exchange from Romeo and Juliet, the fine-tuned chatbot responds in a way that contrasts humorously with the second non-chatbot conversant while answering the firsts' question.

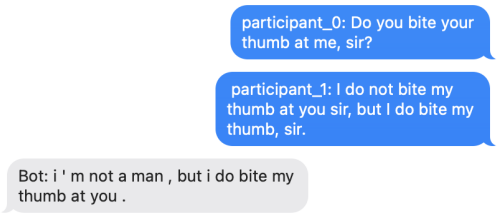

Figure 3: **Fine-tuned BlenderBot Multi-Party Conversation**. The tuned chatbot is on the left in gray, and the non-chatbot conversants are on the right in blue.

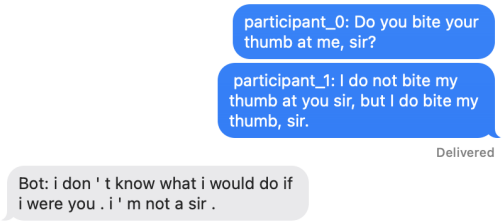

Figure 4: **Unfine-tuned BlenderBot Multi-party Conversation**. The unfine-tuned chatbot is on the left in gray, and two different non-chatbot conversants are on the right in blue.

6 Analysis

We begin by understanding the strengths and weaknesses of unfine-tuned BlenderBot when it is newly placed in the multi-party setting. Figures 6 and 7 below show the unfine-tuned model's response for two labeled examples of our multi-party dataset. The labeled examples are generated from the

same raw Ubuntu conversation, with the chatbot replacing first the second speaker then the third as in Figure 2. In the first labeled example, Figure 6, the chatbot gives a clear, committal response to the single other conversant. By contrast, in Figure 7, the chatbot appears to be confused by the second conversant's response. The chatbot reverts to making some disclaimer before giving a generic encouraging suggestion at odds with the second conversant's complaint. In short the unfine-tuned BlenderBot is qualitatively as-yet-unprepared to benefit from seeing input text separated into distinct non-chatbot conversants.

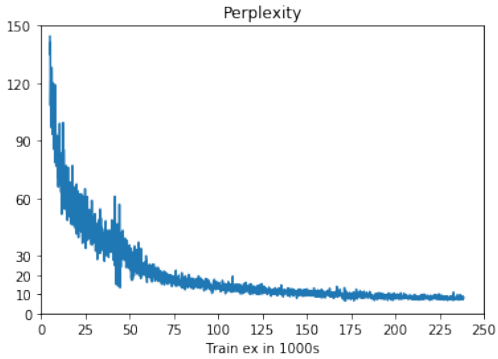

Figure 5: **Fine-tune loss consistently decreasing.** At this scale, perplexity appears to flatline. It decreased consistently until the end of training

One factor which might limit the speed with which fine-tuning improves BlenderBot's performance on multi-party examples is that BlenderBot must 'learn' the importance of the speaker name given in input text. Future work could explore whether BlenderBot improves faster if it is given a more explicit representation of different non-chatbot conversants. For example, BlenderBot is already configured to update a history object across examples in a single episode. Changing the architecture to maintain a history object for each conversant (chatbot and non-chatbot) might help the model ascribe importance to the difference in speakers faster.

Figure 6: **Unfine-tuned BlenderBot Two-party Conversation**. The chatbot is on the left in gray, and the non-chatbot conversant is on the right in blue.

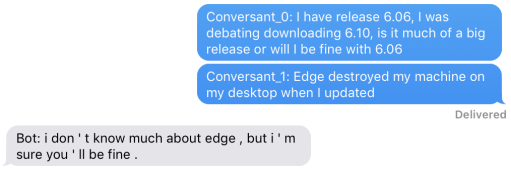

Figure 7: **Unfine-tuned BlenderBot Multi-party Conversation**. The chatbot is on the left in gray, and two different non-chatbot conversants are on the right in blue.

From the outset, our experimental design had the inevitable drawback that fine-tuning BlenderBot on a dataset different from any it had been trained on could reasonably be expected to improve its performance in the multi-party setting even if we had input the data as two-party episodes instead of multi-party. We implemented an initial exploration of this question by creating a dataset of "obscured multi-party conversations" via a one-to-one map from the fine-tuning dataset we engineered. The training examples in this dataset were "obscured" in the sense that all non-chatbot text was presented as if being generated all at once by a single non-chatbot conversant. This is essentially how chatbots configured for two-party conversation are forced to represent multi-party conversations. Across the first 1000 training examples, our model's loss was comparable to the loss of the model trained on "obscured" data. However, if "obscured" training data is genuinely less expressive, we might expect this to only manifest after significant training, so this question is still undetermined.

7 Conclusion

Between chatbots designed for two-party converserations, ill-suited to multi-party conversations and effective multi-party conversation chatbots which are drastically different in structure from their two-party conversation counterparts, we take an intermediate approach. Our work provides a general, well-motivated way of quantifying the effectiveness in multi-party conversations of a chatbot configured for two-party conversations, and our fine-tuning approach provides a way of improving the multi-party conversation performance of any chatbot.

It would be natural for future work to begin by fine-tuning longer to obtain lower perplexity and better ACUTE-EVAL. After success in this endeavor, to evaluate how much of the success is attributable solely to the new data (as discussed in the previous section) one could compare the following two models' performance in the multi-party setting: BlenderBot fine-tuned on Ubuntu data which is input as multi-party data and BlenderBot fine-tuned on Ubuntu data which is input as two-party data.

References

[1] Sujith Kochupurackamuriyil Sam. Opportunities for multi party chatbots in human social interactions. *MAI 82/1(E), Masters Abstracts International*, 2020.

[2] Seokhwan Kim, Michel Galley, Chulaka Gunasekara, Sungjin Lee, Adam Atkinson, Baolin Peng, Hannes Schulz, Jianfeng Gao, Jinchao Li, Mahmoud Adada, et al. The eighth dialog system technology challenge. *arXiv preprint arXiv:1911.06394*, 2019.

[3] Weishi Wang, Shafiq Joty, and Steven CH Hoi. Response selection for multi-party conversations with dynamic topic tracking. *arXiv preprint arXiv:2010.07785*, 2020.

[4] Geoff Kaufman Jessica Hammer Joseph Seering, Michal Luria. Beyond dyadic interactions: Considering chatbots as community members. *Proceedings of CHI*, 2019.

[5] The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems.

[6] Naman Goyal Da Ju Mary Williamson Yinhan Liu Jing Xu Myle Ott Kurt Shuster Eric M. Smith Y-Lan Boureau Jason Weston Stephen Roller, Emily Dinan. Recipes for building an open-domain chatbot. 2020.

[7] Jonathan K. Kummerfeld, Sai R. Gouravajhala, Joseph Peper, Vignesh Athreya, Chulaka Gunasekara, Jatin Ganhotra, Siva Sankalp Patel, Lazaros Polymenakos, and Walter S. Lasecki. A large-scale corpus for conversation disentanglement. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, July 2019.

[8] Stephen Roller Margaret Li, Jason Weston. Acute-eval: Improved dialogue evaluation with optimized questions and multi-turn comparisons. *Arxiv*, 2019.