Predicting Backchannels In Conversations

Independent Study advised by Professor Lyle Ungar

Ishita Agarwal
School of Engineering and Applied Science
University of Pennsylvania

Abstract

This paper explores the feasibility of incorporating backchanneling into computer-mediated communication. Utilizing the CANDOR corpus, we investigate the characteristics of backchanneling. We present a machine learning model designed to predict the occurrence of backchannel initiations in conversations between two people. We also explore potential correlations between backchannel rates and speaker characteristics. Additionally, we propose a human-experience metric that better captures the accuracy of backchannel prediction. Our findings not only enhance our understanding of conversational dynamics but also pave the way for developing more interactive and responsive chatbots. This study represents a step toward bridging the gap between human-human and human-computer communication by enabling chatbots to engage in more human-like, interactive dialogue through generative backchannel models.

1 Introduction

In human-computer conversations, currently the computer does not backchannel or interrupt while a human is speaking. It only begins its response when the speaking partner stops. However, in normal human communication, backchanneling and interruptions are common, providing immediate feedback to the speaker. This semester, we aim to explore the following questions: How can we define backchanneling in a conversation? How do the amount, frequency, and positioning of backchanneling and interruptions vary as a function of age, gender, and region?

For this project, we will use the CANDOR corpus, which is a repository of conversations between humans in the form of audio, video, and transcripts. We have developed a logistic regression model that performs binary classification to determine whether the last word in a sentence segment is a backchannel. Additionally, we have established a human-experience metric to more effectively evaluate the accuracy of backchannel positioning.

2 Motivation

Backchannels are essential for providing instant feedback in conversations, yet current chatbots lack the capability to backchannel or interrupt while a human is speaking. Introducing the ability for chatbots to backchannel at appropriate times could make interactions feel more natural. The development of a model capable of predicting the likelihood of a backchannel during an ongoing conversation represents the initial step toward creating a generative model for backchanneling. Additionally, investigating whether speaker characteristics like age, gender, and regional background correlate significantly with backchanneling rates could offer deeper insights into improving conversational models.

3 Background and Related Work

The term ‘backchannel’ was first introduced by Victor Yngve in 1970 to describe the subtle, often non-verbal responses that listeners provide during a conversation. Since then, various studies have expanded on this concept. For instance, the paper titled ‘Automatic Detection of Discourse Structure for Speech Recognition and Understanding’ (Jurafsky et al., 1997) delves into the dynamics of backchannel opportunities in speech, highlighting the importance of timing and contextual appropriateness in generating these cues.

A significant recent contribution is the research paper "Turn-Taking and Backchannel Prediction with Acoustic and Large Language Model Fusion" by Wang et al. (2024). This study explores advanced techniques for predicting backchannel moments by combining acoustic signals with large language models, aiming to enhance conversational AI. The researchers developed a predictive model that classifies turn-related behavior into three categories:

continuing speech, backchannel, and turn-taking, using the Switchboard corpus for training. The textual data is encoded using two large language models, GPT-2 and RedPajama, while HuBERT is employed for encoding audio data.

In contrast to Wang et al.'s approach, our research uses not only the encoded conversational text using RoBERTa but also integrates relevant conversational features such as the backchannel rate so far, the number of words spoken since the last backchannel, and the cumulative number of backchannels. Instead of the Switchboard corpus, we utilize the CANDOR conversational corpus. While Switchboard relies on manual transcription, CANDOR uses automated transcription via the AWS Transcribe API, aligning more closely with our goal of developing generative backchannel models. We opt to focus solely on textual features, omitting audio signals based on findings that show only a marginal difference in classification outcomes with or without audio features. Additionally, we propose a new, more relevant user-experience metric that could more accurately characterize models predicting the initiation of a backchannel.

4 Dataset

The CANDOR conversational corpus, also known as 'Conversation: A Naturalistic Dataset of Online Recordings', is a comprehensive multimodal dataset of human conversations. This corpus includes 1,656 recorded conversations, amounting to over 7 million words and 850 hours of audio and video content. The data set captures a range of vocal, facial, and semantic expressions and includes extensive post-conversation reflections from the speakers.

We have chosen to use the CANDOR corpus for training, classifying, and testing our predictive model over other more widely used datasets such as Switchboard for several strategic reasons. In Switchboard, conversations are manually transcribed, while CANDOR utilizes the AWS Transcribe API for automated transcription. As we plan to develop a generative backchannel model that will handle on-the-fly conversations, relying on an automated transcription service is essential as manual transcription would not be feasible in online scenarios. Training our model on datasets that are transcribed in a manner similar to our intended application environment ensures greater relevance and potential accuracy.

Additionally, CANDOR is preferred for its sophisticated turn models, which are crucial for our specific need to recognize backchannels in conversations. CANDOR defines turn models as algorithms that segment a continuous stream of dialogue between two people into meaningful turns. We are particularly leveraging two turn models from CANDOR, Audiophile and Backbiter, to generate our datasets. Audiophile is a basic model where a turn is defined as the speech segment of one speaker until another speaker interjects, marking the beginning of the new speaker's turn. On the other hand, Backbiter is a more complex turn model that captures not only the words of the speaker but also any backchannel words uttered by the listener during the speaker's turn. This feature is invaluable for our project, as it allows us to precisely determine when backchannels start, aligning perfectly with our project's objectives.

5 Methods

The main goal of this study is to create a model that can accurately predict the beginning of a backchannel in a conversation between two people. The input to the predictive model is conversational data in the form of turns and backchannel words contained within those turns. The expected output is: For a given input sentence (may be a partial utterance) predict if the next word is the beginning of a backchannel (yes/no). We are predicting if at the next word uttered by a speaker, if the listener is going to backchannel or not.

5.1 Dataset Creation

Our dataset for training and testing the predictive model is created based on a combination of both Audiophile and Backbiter turn models. For each conversation ID within the specified range (except for IDs explicitly excluded due to missing information), the process involves the following steps for generating features for analyzing conversations:

1. Load two datasets (backbiter.csv and audiophile.csv) which contain conversation data, including who speaks each line (speaker), the text of what they said (utterance) along with timestamps of each turn.

2. Data Preparation: For each conversation, data frames for both 'backbiter' and 'audiophile' are filtered by conversation ID and sorted by turn ID.

3. Identify the speaker and the listener (backchanneler). They are assumed to be consistent throughout the conversation.

4. Iterate over each line of the backbiter conversation, categorizing utterances into those with and without backchannels based on whether the interval column is filled.

5. For utterances with backchannels, further filter and categorize backchannel utterances from the audiophile data with the specified start and stop times, marking utterances by the backchanneler with special syntax to indicate backchanneling. As a result, a backbiter speaker turn now has backchannel spoken by the listener interjected into the conversation with a special marker '<' that marks beginning of that backchannel, and '>' marks the end of the backchannel utterance. An example of turn slices and interjected backchannels can be seen in Figure 1.

6. Then, feature extraction is done by converting utterances into slices of words for more granular analysis. Therefore, each turn (which consists of a string of words) is divided into slices, such that the number of slices for each turn is equal to the number of words in that turn.

7. Calculate several metrics such as the number of words spoken so far, number of words since the last backchannel, and backchannel count, as well as backchannel rates across the conversation. The label for each slice (whether the last word uttered was the start of a backchannel) is also captured here.

8. At the end, we store all the extracted features and labels into respective directories for each conversation ID for later retrieval or analysis.

5.2 Feature Creation

We generate embeddings for segments of dialogue turns, excluding any backchannel tokens, using RoBERTa (specifically the RobertaModel from the Huggingface transformers library). To reduce dimensionality, we employ PCA from the sklearn library, which reduces the embedding dimensions to 50 features. These features are subsequently incorporated into our model along with the following custom features for each word-by-word subsequence within a backbiter utterance:

Isn't
Isn't how
Isn't how <Mhm.>
Isn't how <Mhm.> some
Isn't how <Mhm.> some of
Isn't how <Mhm.> some of this
Isn't how <Mhm.> some of this stuff
Isn't how <Mhm.> some of this stuff is
Isn't how <Mhm.> some of this stuff is so
Isn't how <Mhm.> some of this stuff is so <Right.>
Isn't how <Mhm.> some of this stuff is so <Right.> expensive.
Isn't how <Mhm.> some of this stuff is so <Right.> expensive. I
Isn't how <Mhm.> some of this stuff is so <Right.> expensive. I spend

Figure 1: Here, each turn is divided into multiple slices which consist of one word more than the previous slice. The backchannels are interjected into the conversations beginning and ending with angular brackets.

1. Embedding of non-backchannel words spoken up to the current point in the backbiter utterance.

2. Number of words since the last backchannel.

3. Total number of words spoken up to the current point.

4. Count of backchannels observed up to the current point.

5. Backchannel rate in the current turn.

6. Overall backchannel rate for the conversation.

7. Slice ID.

8. Turn ID.

The label is assigned a value of 1 or 0 to indicate whether the final word of a particular turn slice is a backchannel (1) or not a backchannel (0).

5.3 Classifier

For classification, we utilize the Logistic Regression binary classifier from the `sklearn` library, employing L2 regularization. The penalty parameter C is set to 1.0, and the tolerance for the optimization algorithm is set at 1×10^{-2}. Additionally, we adjust the classification threshold to 0.07 to enhance sensitivity to the positive class.

To prevent overfitting and preserve the integrity of conversational data, we split datasets based on entire conversations. Each conversation, encompassing all turn slices associated with a particular

conversation ID, is exclusively assigned to either the training set or the testing set.

We evaluate the performance of our model using 5-fold cross-validation, comparing accuracy, precision, recall and F1 measure between default metrics and our proposed human-experience metric.

Hello how are you <hey!> nice to meet you

Figure 2: The proposed human-experience metric. The true label position of the backchannel is specified at the start of the angular brackets. Our metric proposes that if the predicted start of backchannel is before or after any word in the dotted box, the prediction still counts as a true positive.

5.4 Human-Experience Metric

We propose a human-experience user experience metric for determining true positives in the predicted values. Along with the default case where the predicted value equals the label value for a prediction, we consider an additional scenario. If the predicted label is true (i.e., predicted start of backchannel), but the true label is false (i.e., no backchannel actually started at this point), we do not mark it as a false positive. Instead, we assess whether the previous two slices or the next two slices in the same turn, within the same conversation (if they exist), have a true label value. If yes, we mark that prediction as a true positive; otherwise, it is a false positive. This approach indicates that our model was still able to predict the rough location of the beginning of the backchannel. An example can be seen in Figure 2.

We believe this human-experience metric captures the prediction of the beginning of a backchannel more realistically in a way that matters more to the human listener. The speaker wouldn't care much if the listener started to backchannel slightly before or after the speaker expected, as long as the rough starting point of the backchannel is correct. We believe that the human-experience user experience metric allows us to capture information that would otherwise be lost in standard accuracy metrics.

6 Results

6.1 Predictive Model

We obtain an average AUC of 0.789, as can be seen in Figure 3. We also measure the performance of our model based on the human-experience accuracy metric as defined in Methods. Using the human-experience accuracy metric significantly alters the performance evaluation of the model, generally leading to higher values in accuracy, precision, recall, and F1 scores. There is a consistent improvement in F1 Score with human-experience metrics, reflecting a better balance between precision and recall, as can be seen in Figure 4.

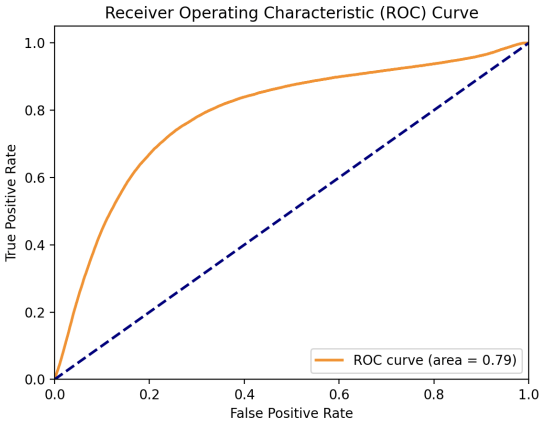

Figure 3: ROC Curve for the model with default accuracy metrics.

Metric	Without Custom Metrics	With Custom Metrics
Accuracy	0.8562	0.8864
Precision	0.1733	0.3810
Recall	0.5162	0.7011
F1 Score	0.2595	0.4937

Figure 4: Comparison of Model Performance Metrics with K-Fold Cross-Validation with and without human-experience metrics. The human-experience evaluation metric indicates that our model was still able to predict the rough location (at most 2 words misplaced on either side of the true label) of the beginning of the backchannel, which the default evaluation metric fails to capture.

6.2 Correlation Study

We used Spearman correlation for testing for significance to find correlation between speaker features such as age and backchannel rate. However, we

were not able to find any statistically significant correlations.

7 Conclusion and Future Steps

In our work, we created a predictive model that predicts the beginning of a backchannel in a conversation between 2 people with an average AUC of 0.789. We also proposed a novel method of evaluating performance of this predictive model that more accurately captures the human user experience. We also attempted to find correlations between speaker features and backchannel rate, however we were unable to find anything statistically significant.
As a future step, we would attempt to generate backchannels and interruptions as part of a human-computer conversation. We would also like to evaluate the human-experience metric through user studies. In the long term, this study will help us understand when a human is conversing with a computer, if generated backchanneling/interruptions from the computer helps improve the user experience and problem solving ability of the pair.

The code for the GitHub Repository can be found here.

References

[1] Yngve, Victor. *On getting a word in edgewise*, in the Sixth Regional Meeting of the Chicago Linguistic Society, 1970.

[2] Jurafsky, D., Shriberg, E., Fox, B., & Curl, T. *Automatic Detection of Discourse Structure for Speech Recognition and Understanding*, in Speech Communication, Vol. 22, No. 2, pp. 127-132, June 1997.

[3] Wang, et al. *Turn-Taking and Backchannel Prediction with Acoustic and Large Language Model Fusion*, to appear in Proceedings of the Conference on Computational Linguistics, 2024.

[4] Reece, et al. *The CANDOR Corpus: Insights from a Large Multimodal Dataset of Naturalistic Conversation*, in Science Advances, 2023. DOI: 10.1126/sciadv.adf3197