オープンプラットフォームとクラウドソーシングを活用した対話コーパス構築方法

塚原 裕史　　内海 慶
株式会社デンソーアイティーラボラトリ 研究開発グループ
{htsukahara, kuchiumi}@d-itlab.co.jp

1 はじめに

最近のスマートフォンなどにおけるクラウドを利用した音声認識の精度向上によって、音声対話エージェントのサービスの利用が、一般に普及し始めている。これらのサービスでは、情報検索などのスマートフォンの機能を音声で簡単に呼び出すことを目的としており、できるだけ一問一答で簡潔に対話が完了するように設計されている。このようなタスク指向の対話のみではなく、人と雑談を楽しむというような非タスク指向の対話が行えるような対話エージェントに対する期待も高まっている。もし、これが実現すれば、対話エージェントがより一層、身近なものとなり、日常で定常的に利用され、対話システムの価値が高められると考えられる。

非タスク指向の対話生成に向けて、ウェブ上のニュース記事、Wikipedia、Twitter やクラウドソーシングを活用する研究がある [6, 5, 2] 。

本論文では、雑談生成を目的とした対話コーパスを効率的に構築するために、オープンプラットフォームのチームコミュニケーション環境で、クラウドソーシングのワーカー同士による対話入力とそのアノテーション付けを同時の行う方法を提案する。従来、クラウドソーシングによるデータ収集では、各ワーカーはお互いに独立に作業を行うが、本提案ではワーカー同士が協力してデータを作成するという点に特徴がある。人同士がコミュニケーションすることで、機械的に成りがちな対話表現ではなく、話題を盛り上げるような発話のパターンを多く収集できると期待できる。

2 背景と課題

従来のタスク指向での対話システムでは、ユーザの発話やシステムの発話形式を規定するテンプレートを予め設計しておき、その設計されたテンプレートベースによる発話のみをやり取りする。また、入力された発話に対する応答も予め設計されたルールに従って、システム応答が決定されるルールベースによる対話制御が行われてきた。したがって、雑談のように色々な話題に対応し、かつ長く対話を持続するには、膨大なルールとテンプレートを設計する必要になるという問題があった。ルールベースによる対話生成が失敗した際に、対話コーパスを利用し、統計的な対話生成を行うという研究がある [7]。

しかし、対話コーパスの構築には、一般に対話音声の録音、対話文の書き起こし、アノテーション付けなどの作業が必要であり、統計的な手法が適用できるような大規模な対話コーパスを構築するには、そのコストも問題となる。ウェブ上で活用できる演劇用の台本のような対話データもあるが、標準的なデータ形式がなく、機械的な処理に向かず、かつ統計的な手法を適用するには、規模が小さく、また雑談をするには、話題が十分ではないなどの問題がある。また、掲示板や Twitter などのデータをクロールして利用する研究もあるが [2]、商用のサービスへ応用するには、著作権やプライバシーに関わる問題などを解決しなければならない。

本論文では、このような課題を鑑みて、クラウドソーシングとチームコミュニケーションプラットフォームを活用した対話データ収集方法を提案する。クラウドソーシングは、近年、機械学習用のデータ作成などに盛んに活用されるようになっているサービスであり、多くのワーカーが独立に並列的に作業を行うことで、一度に大量のデータを構築できるメリットがある。チームコミュニケーションプラットフォームも最近ソフトウェアの開発などの現場において盛んに利用されるようになっており、Slack[1], Yammar[2] などのいくつかの主要なサービスがある。これらでは、主にチャット形

[1]https://slack.com/
[2]https://www.yammer.com/

式の対話によってワーカー間の意思疎通を活発化することに利用されるが、外部のクラウドサービスとも連携できるカスタマイズ機能が充実している。我々は、話者をクラウドソーシングし、チームコミュニケーション環境において、対話を行ったログを収集する方法を提案する。

このような方法には、以下のようなメリットがあると考えられる。

- チャット形式であり、書き起こしのコストが要らない。また、すべての発話は、自動的に一意的なIDが付与され区別することができる。
- 相手の応答待ち時間を利用し、自身の発話入力後に、その発話内容へのアノテーション付けを同時に行いながら対話できる。（アノテーション付けは、メッセージの修正機能を利用して行う。）また、アノテーション用のマーカーをカスタマイズすることで、入力補完機能を利用して、正確かつ効率の良いアノテーション付けができる。
- チーム内での対話ログはREST形式などのAPIでアクセスすることができる。
- API経由でユーザが入力した発話内容を外部で処理して、作業の進行管理やアノテーション付けのサジェストや修正などのサポートができる。

従来からあるSkypeなどのサービスもあるが、各ユーザのローカル環境にアプリケーションをインストールしなければならず、かつ、対話のログがローカルファイルとして残されるため、対話コーパスを構築するリモートの開発者から管理できないという問題があることに注意する。

3　関連研究

人による対話の場合、同一の内容であっても、会話の流れや話者に癖や個性によって様々な表現がなされ、そのようなことが、人らしい自然な対話を行ったり、雑談のような長い対話を飽きずに行う上で、大切であると考えられる。テンプレートベースでの対話生成において、同一の内容に対する多様な表現パターンを獲得するために、クラウドソーシングを利用する研究がある [4]。

また、独自のチャットシステムを構築し、その上にクラウドソーシングからワーカーの組を割当、所定のタスクを協力して解決するための対話を収集するという研究がある [3]。この研究は、我々の提案する対話収集に非常に近いが、彼らの研究はタスク指向の対話であり、比較的短い対話のやり取りで目的が達成されて終了しており、雑談のような非タスク指向の対話を生成する目的にはそぐわない。また、我々はワーカー自身が対話入力と同時に、アノテーション付けと校正作業を効率的に作業できるような仕組みを独自のシステムを構築するというよりも、既存のプラットフォームと外部サービスとの連携をカスタマイズするレベルで実現している。

一方、クラウドワーカーをリアルタイムで利用し、Twitterのメッセージログからリツートで連接した対話ペアを収集し、データベースを構築し、ユーザからの入力発話と類似した発話ペアの入力を持つ対話ペアを検索し、応答を生成し、そのような対話ペアが見つからない場合にクラウドソーシングで、ワーカーにリアルタイムで発話文を収集して返すという研究がある [1]。

4　対話データ収集方法

図1に、我々が提案するクラウド上のチームコミュニケーションプラットフォームと外部サービスとの連携によるクラウドソーシングのワーカー間の対話環境の概要を示す。図中の番号は、作業の流れを示している。

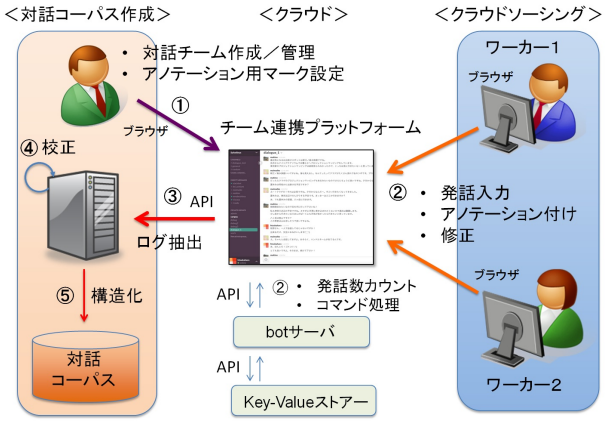

図 1: クラウドソーシングによる対話データ収集環境

まず、①では対話コーパス開発者が、チームを作成し、外部連携機能やアノテーション付け用のマーカーなどのカスタマイズを行う。また、対話セッション毎にチャンネルを作成し、アクセス用のURLを割り当てたワーカーへメールする。②では、ワーカーはブラウザにより、指定されたチャンネルのURLへアクセスし、対話とアノテーションを入力する。入力された

対話データは外部連携サービスに転送され、ワーカー毎に発話数をカウントしたり、メッセージ内に埋め込まれたコマンドの処理、アノテーション候補のサジェストなどの処理を行い、またそれらの処理結果は、bot 連携機能によりチャンネル上に適宜、リアルタイムに通知する。例えば、所定の発話数が収集できると、ワーカーに対して作業完了がチャンネル上に bot により自動的に通知される。対話入力とが完了後、③では、対話ログを取得し、不正なアノテーションのチェックと補正とアノテーションの校正用のデータファイルを生成する。④では、校正者は、校正用データファイル上で、修正が必要な箇所を添削する。校正者は複数人で行い、それらの結果を多数決などによって統合する。アノテーションの添削結果は、ワーカー毎に分割され、各ファイルはチャンネルへアップロードすることで、修正指示が通知される。ワーカーによる修正結果の確認は、添削ファイルと現在の対話ログとの差分を取ることで自動チェックすることができる。差分が無くなるまで、②、③、④のプロセスが繰り返される。図 2 に、その詳細な手順を示す。これらの作業が完了後、⑤では、アノテーションデータを利用して、構造化された対話コーパスへマージする。

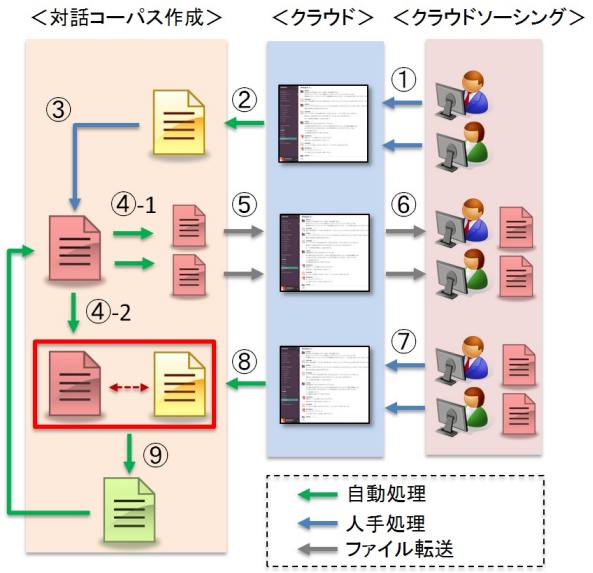

図 2: アノテーション校正の作業手順

5　対話データ収集

5.1　実験システム

今回、提案方法を実施するために、チームコミュニケーションプラットフォームとして、Slack を利用した。また、外部連携として Heroku[3] 上に Hubot[4] を利用した bot システムを用意し、チャンネル上でのワーカーの発話数のカウント及び発話としてカウントされないコメントやコマンドの処理と作業進捗の通知を行うようにした。また、これらの対話ログ以外の処理データを管理するために、Heroku のアドオンである Redis cloud という KVS 形式のデータベースを利用した。slack 上の対話ログは、REST 形式の API を利用して抽出し、各発話毎に添削用の行などを追加して添削用ファイルを生成できるようにした。添削用データの各ワーカー毎への分割、添削内容と修正結果との差分のチェックには、それぞれ簡単な Ruby スクリプトを用意した。各発話は、自動的に ID が振られているため、添削箇所のワーカーへの指示や差分のチェックなどの自動化は非常に容易にできる。以上は、すべてオープンなシステムを利用しており、簡単なスクリプトをいくつか作るのみで、それらをカスタマイズし、実現できている。また、上記のサービスはミニマムの構成であれば、現時点では、すべて無料から利用開始でき、非常に低コストである。

5.2　アノテーション

今回、図 3 に示すような、発話内のキーワード（固有表現）、話題及び対話行為のアノテーション付けを行うようにした。これらは、ワーカーが対話入力と同時にでも入力できるような簡略された形式にしてある。例えば、対話行為の様に、予め決められたものについては、カスタムのマーカーを作成し、アノテーション用文字列の最初の数文字を入力すると、タブを押す事で、選択候補が入力欄にポップアップされ、さらにタブで選択することができるようにした。このような補完機能が利用できることで、作業効率アップとタイプミスの抑制が期待できる。

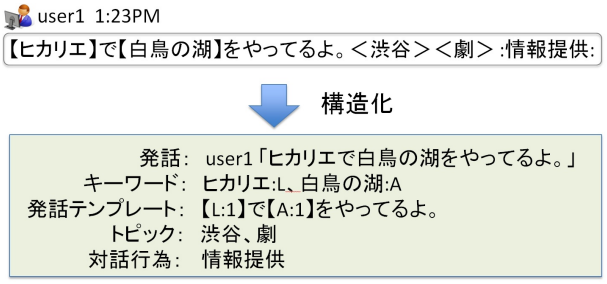

図 3: アノテーションと対話データの構造化

また、対話入力は一度に

[3]https://www.heroku.com/
[4]https://hubot.github.com/

5.3 データ集種

対話生成への利用を考えて、個々の発話はできるだけ直前の発話のみに依存し、それよりも先の長い文脈に依存しないように入力するなどの作業ガイドラインを設定した。対話入力は、2 人一組として、1 日当り 2 ～ 3 組で入力を行った。1 回の作業単位は、新しいチャンネルを作成し、お互い 100 発話入力するように設定し、入力が完了すると bot が自動的に通知するようにした。1 発話の入力には、平均 2 分ほど要し、100 発話の入力に、通常 3 ～ 4 時間を要した。話題は、スポーツやグルメなどのニュース記事に関してお互いに話すようにした。”rem” と先頭に入力した発話は、対話データとして収集しないコメント用の発話とし、ワーカー同士での休憩時間の設定などの作業調整のやり取りをチャンネル上で出来るようにした。また、対話コーパスの開発者からも、ワーカーへの指示をコメントを利用し、チャンネル上でスムーズに作業を進めることができた。休憩を入れた場合には、また新しい対話セッションを開始するということで、”NewDial” という制御用の発話を用意した。コーパス構築時には、一つのチャンネルをこの制御用の発話を元に、いくつかのセッションに分割した。アノテーション校正は、まずはワーカー自身での第 1 のチェックを行い、対話コーパス作成側で、2 名が校正を行って最終的に決定した。約 2 ヶ月間で、100 チャンネル分収集し、約 2 万発話のアノテーション付きデータが収集できた。

6 データ分析

今回、チャット形式による対話では、できるだけ文脈に依存せず個々の発話毎に意味が通じるように、というガイドラインを設定したため、応答時間が約 2 分ほど掛かり、この間に相手が応答を待たずに、別の質問を入力してしまい、このような場合、図 4 に示すように、直前の質問を答える前に、準備していた応答を入力してから、続けて質問への回答を入力するというような本来の対話での発話の順序が逆転するような現象がしばしば見られた。このような問題については、対話行為のアノテーションで逆転現象を検出し、自動的に順序を補正するという対処を考えている。

7 おわりに

本論文では、クラウドソーシングとチームコミュニケーションプラットフォームを活用した対話データ収集方法を提案し、実際に、キーワード、話題、対話行為がアノテーション付けられた対話データを約 2 万発話文ほど収集した。今後は、この対話コーパスでの統計情報の分析や雑談生成アルゴリズムの研究に応用する。

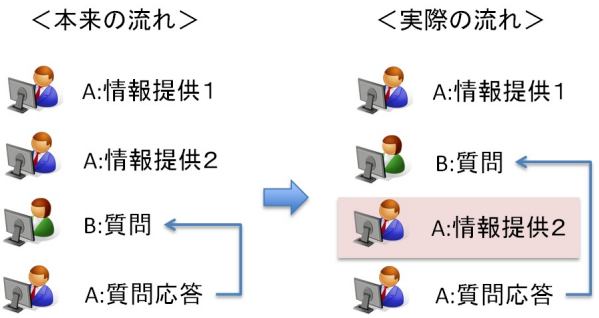

図 4: 発話の前後関係が入れ替わる現象

参考文献

[1] Fumihiro Bessho, Tatsuya Harada, and Yasuo Kuniyoshi. Dialog system using real-time crowdsourcing and twitter large-scale corpus. In *SIGDIAL '12*, pp. 227–231, 2012.

[2] R. Higashinaka, N. Kawamae, K. Sadamitsu, Y. Minami, T. Meguro, K. Dohsaka, and H. Inagaki. Building a conversational model from two-tweets. In *ASRU 2011*, pp. 330–335, 2011.

[3] Walter S. Lasecki, Ece Kamar, Dan Bohusa, and Eric Horvitz. Conversations in the crowd: Collecting data for task-oriented dialog learning. In *Workshop at Conference on Human Computation and Crowdsourcing 2013*, 2013.

[4] Margaret Mitchell, Dan Bohus, and Ece Kamar. *Proceedings of the INLG and SIGDIAL 2014 Joint Session*, chapter Crowdsourcing Language Generation Templates for Dialogue Systems, pp. 172–180. 2014.

[5] Graham Wilcock. *Proceedings of the Workshop on Question Answering for Complex Domains*, chapter WikiTalk: A Spoken Wikipedia-based Open-Domain Knowledge Access System, pp. 57–70. The COLING 2012 Organizing Committee, 2012.

[6] 灘本明代, 林正樹, 道家守, 浜口斉周, 田中克己. 係り受け構造及びシソーラスによる対話文生成と簡易演出技法を用いた web コンテンツの受動的視聴. In *DEWS2005*, 2005.

[7] 豊美目黒, 弘晃杉山, 竜一郎東中. ルールベース発話生成と統計的発話生成の融合に基づく対話システムの構築. 人工知能学会全国大会論文集, Vol. 28, pp. 1–4, 2014.