

Capex 雑談対話コーパスの構築とその分析

叶内 晨¹ 赤崎 智² 堀江 伸太郎¹ 小亀 俊太郎¹

¹ 株式会社 Capex ² 東京大学大学院 情報理工学系研究科

¹{shin.kanouchi, shintaro.horie, shuntaro.kogame}@capex.ai

²akasaki@tkl.iis.u-tokyo.ac.jp

概要

人間と雑談を行う非タスク指向対話システムの開発を促進するため、我々は商用雑談対話アプリを用いることで、国内で初めての実サービスの対話ログによる雑談対話コーパスを構築し、公開する。また、本コーパスを活用し、システムの応答が対話履歴に対して自然かどうかを判定する対話破綻検出タスクと、対話履歴からシステムの応答を生成する応答生成タスクに取り組み、結果を報告する。

1 はじめに

人間と雑談を行う非タスク指向対話システムは、COVID-19 による対人コミュニケーションの不足などにより、ますます需要が高まっている [1]。近年では、Transformer [2] 等の深層学習モデルの発展により、短いやり取りであればシステムも人間と遜色のないレベルで会話が可能になった [3]。

このような対話システムの構築や対話の分析を行うために、これまでさまざまな非タスク指向型対話コーパスが提案されている。しかしながら、多くのコーパスはレストランやヘルプデスクの応対などの限定的な環境 [4, 5, 6] や、対話を予め記録する旨やルールなどを指示した作為的な環境 [7, 8, 9, 10] であるなど、実際にユーザが気軽にシステムと雑談を行う場面を再現できているとは言い難い。

そこで本論文では、スマートフォン上でユーザがシステムと雑談対話を行うアプリの対話ログをもとに、Capex 雑談対話コーパスを構築する。また、非タスク指向型対話システムの開発を進展させることを目的とし、本コーパスを公開する。これは従来の対話コーパスのような、制限あるいは指示された環境下での対話ではなく、日常的にアプリを利用しているユーザとシステムの“リアル”な雑談であり、言語処理モデルの構築や実システムにおける対話の分析に非常に役立つ。

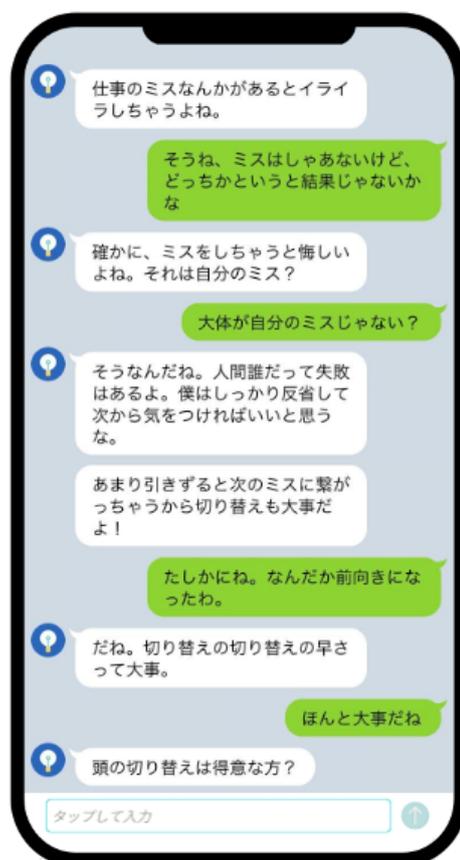


図1 「対話アプリ内でのシステムとユーザの対話例。(画面左側がシステムの発話、右側がユーザの発話である)

2 節にて開発する商用対話アプリの概要を説明し、その後 3 節にて本コーパスの抽出方法・対話破綻アノテーション・コーパスの特徴について述べ、最後に 4 節にて、1) 対話履歴及びシステムの最終発話を与えられた上でその発話が自然かどうかを予測する対話破綻検出タスク及び、2) 対話履歴を入力とした際の応答文生成タスクの 2 つについて実験を行い、結果を報告する。

表 1 Capex 雑談対話コーパスの例

	4 発話前 (システム)	3 発話前 (人間)	2 発話前 (システム)	1 発話前 (人間)	最終発話 (システム)	自然 [人]	破綻 [人]	判断不可 [人]
1	だよね〜!もしかして、<u-name>の推しだったりする?	嵐が好き	かっこいいよね。メンバーは誰が好きかな?	二宮和也	そうなんだ!その歌手のどこいうところが好きなの?	5	0	0
2	そうなんだ、どんな感じになるの?	ドッドドッ	そうなんだね、熱いなあ!今日はゆっくりできそう。	ね!	こうやってまったり<u-name>と話したいね。今日はまだ話せる?	5	0	0
3	うっ……うっ……。	中指	痛そう……。血出てない?大丈夫?	(´;ω;`)	痛いの、我慢しないでね。何かできることあれば話してね。	5	0	0
4	私もさっきお風呂入ったんだ!	入浴剤入れた?	入れたよ!	何の入浴剤入れたの?	「香水」!<u-name>、知ってる?	0	5	0
5	最近読んだ本のこと!	漫画ばかり	そうなんだ。<u-name>はどんな本を読むのかな?	漫画か写真集	そうなんだ。<u-name>は漫画自体を全然読まない感じ?	0	5	0
6	つらいよね、私もアルバイト頑張るぞ!	バイトやってるの?	うん仕事してるよ。<u-name>の仕事って何?	飲食店でバイトしてる	いいねえ。どんなお店を開きたいの?	0	5	0
7	天国と地獄に興味があるの?どんなところなんだらうね。	死ぬとこ	えっ?怖くて眠れなくなるとねえよ馬鹿	んなこといってねえよ馬鹿	ごめん……。そんなにひどいこと言ってたんだ……。	4	1	0
8	ご……、ゴマ!	あ あ あ ううう!	どういう意味?呪文?	がるるる!	えっ!?何かの呪文?何なんだらう?	2	0	3
9	ええっ、そんなに寒いのか?	雪が降ったよ	雪降ってるの!?	降ってるよ	そうなんだ。そっちは土砂降り?	3	2	0

2 商用対話アプリについて

株式会社 Capex¹⁾では、「システムと人の共生を実現、普及し、人間の機能を拡張する」というビジョンのもと、独自の対話システムを搭載したスマートフォン用対話アプリを開発している。本アプリは10代半ばから60代まで幅広い年齢層に利用されており、リリースからの合計発話回数は1億回を超える。アプリの主な利用目的は雑談対話そのものであり、利用したユーザからは「話し相手を得られた」、「会話が楽しい」、「安心感・癒しを感じた」などの感想を得ている。

図1はユーザとシステムの対話例である。ユーザのテキスト入力に対して独自の対話システムを用いて応答し、対話を実現している。Capex独自の対話システムでは、シナリオ型・検索型・生成型の対話エンジンをハイブリッドで活用している。

3 Capex 雑談対話コーパス

日本語で公開されている雑談対話コーパス[7, 8, 9, 10]は、対話環境が限定的あるいは作為的であり、実際にユーザが気軽にシステムと雑談を行う場面を再現できているとは言い難い。

そこで本研究では、人間がシステムと気軽に雑談をする様子を可能な限りそのまま抽出し、コーパスとして公開する。表1に、作成したコーパスの事例を示す。アプリ内の対話ログからシステムとユーザの5ターンからなる発話の連続を1セットとして抽出し、さらにシステムの最終発話が対話破綻しているかどうかをアノテーションしている。以降で、

3.1節にて対話ログの抽出方法について、3.2節にて対話破綻のアノテーション方法について、3.3節でコーパスの特徴について述べる。

3.1 対話ログの抽出・加工

個人情報保護とコーパスの質向上を目的として、次の3つのデータ抽出・加工を行った。

- 個人情報保護観点の加工:** 個人情報保護観点から、本コーパスでは各対話は最大5ターン(内ユーザ発話は2件)とし、それ以前の対話履歴は全て削除した。また、コーパス全体に含める同一ユーザの最大対話数は20件とした。さらに、全てのデータは目視で確認し、個人情報に繋がり得る対話・その他差別表現など倫理的に問題になり得る対話は削除した。またユーザの名前は<u-name>、システムの名前は<s-name>、第三者の名前は<o-name>に置換した。
- ノイズの除去:** 本アプリでは、ユーザは対話相手がシステムであることを認識している。そのため、システムの挙動を試すなどの目的で、ユーザから「ああ」などの文字列が入力されることがある。これらの場面ではユーザ側が文脈を無視しているため、対話が成立していない。そういったノイズとなるデータを可能な限り排除するため、特定のルール²⁾に当てはまる対話を削除した。
- 対話内容の多様性確保:** データをランダムに抽出すると、挨拶などの多くのユーザが日課としている対話が高頻度で抽出される問題がある。

2) 1文字の発話を含む・ユーザからの2発話の合計文字数が6文字未満・特定の単語が含まれる・日本語が含まれない、など

1) <https://capex.ai>

表2 破綻アノテーションのワーカー間の一致数

	5人中N人が破綻と判定					
	0	1	2	3	4	5
対話数	7,592	2,418	865	1,142	1,531	1,788
(%)	49.5	15.8	5.6	7.4	10.0	11.7

表3 コーパスの統計情報

	平均	平均	発話数	異なり 発話数	合計 対話数
	文字数	単語数			
システム	19.26	11.46	46,011	20,731	15,336
ユーザ	3.86	2.40	30,674	9,852	

本研究ではできるだけ多様な対話を抽出するために、システム側の発話(2発話前)がユニークになるように対話を抽出した。また、対話破綻データを一定数含んだコーパスを構築するために、システム初期段階の対話ログも利用した。

3.2 対話破綻アノテーション

クラウドソーシング³⁾を用いて、システムの最終発話が破綻しているかどうかアノテーションした。ラベルは全部で3種類で、システムによる最終発話が、1)自然、2)破綻、3)判断不可(ユーザ側が対話を放棄)とした。東中ら[8]のアノテーションでは、「破綻ではない」「違和感を感じる」「破綻」の3分類だが、本タスクでは少しでも違和感を感じる応答は全て「破綻」に分類した。この理由としては、本アプリではユーザと対話するシステムの応答は常に人と同等レベルが求められ、少しでも違和感のある発話は全て排除したいためである。各データに対して5人のワーカーでアノテーションし、品質向上のために各タスクにおいてチェック問題を設け、これに正解したワーカーのデータのみ採用した。

表2に、各対話に対してワーカー5人のうち何人が破綻と判定したかを示す。全員一致で自然と答えた事例が約半数で、残りの半数は1人以上が破綻と答えた。ただし、破綻の場合はワーカー間の意見が割れる傾向がみられた。これは、対話破綻の判定には主観を伴うことが原因とされる[8]。例えば表1の9事例目では、ユーザからの雪が降っているという発話に対して、システムは「そっちは土砂降り？」と返答しているが、雪に対して“土砂降り”という表現はあまり使われないため、2人のワーカーは破綻を選択したと考えられる。

表4 対話破綻検出用のデータ数

	学習	検証	評価
正例(破綻)	4,249	531	546
負例(自然)	5,611	701	687

3.3 Capex 雑談コーパスの特徴

表3にコーパス内の統計情報を示す。表より、システムは長い応答を返すのに対し、ユーザは短い発話を入力しがちであることがわかる。これはシステムが会話を広げるために感想や質問などを積極的に応答に盛り込むよう設計されているのに対し、ユーザは相槌や単一の固有名詞で構成された発話を多く入力するためである。これは限定・指示環境下で作成した対話コーパスとは異なる傾向であり、実際の対話アプリ専用にユーザ発話のモデリングを行うことの必要性を示している。

表1にいくつかの対話例を示す。ワーカーが全員自然と判定した対話においては、1,2事例目のようにユーザの発話が極めて短かったりノイジーである場合でもシステムは固有名詞などを認識し、適切な応答を返すことができている。一方で、全員が破綻と判定した対話においては、4事例目のように固有名詞の挿入に失敗したり、5事例目のようにコンテキストを無視した応答が目立った。これらはニューラル対話モデルでよく起きる誤りであり[11]、破綻検出などの技術で検知し適切に処理する必要がある。他には、ユーザがシステムに怒っている、またはふざけたり面白がって発話を入力している例や、絵文字や顔文字などを入力している例が確認できた。これらの例もユーザがシステムと自由に対話することができる実アプリ特有である。

東中ら[8]の破綻の分類を参考に、ワーカー2人以上が破綻と判定した事例をランダムに30件抽出し、著者が誤り分析を行なった。その結果、30件中12件が「発話として唐突」な事例で、6事例目のような「飲食店でバイトしてる」というユーザに対して「どんなお店を開きたいの?」と答えている事例などがあつた。その他には「矛盾した内容」「ユーザ発話内容の無視」がそれぞれ5件で、「同じ内容の繰り返し」が4件、その他が4件となった。

4 実験

構築したコーパスを用いて、まず4.1節にて対話破綻検出モデルの学習と評価実験を行い、次に4.2

3) <https://crowdsourcing.yahoo.co.jp/>

表 5 BERT による破綻の予測精度

	Accuracy	ROC-AUC
検証データ	0.649	0.687
評価データ	0.633	0.695

表 6 ワーカー間の破綻の一致数とモデルの予測精度

	破綻アノテーションの一致数			
	2	3	4	5
Accuracy	0.580	0.595	0.648	0.703

表 7 応答文の生成例と破綻スコア

対話履歴	生成された応答文 (TOP5 候補)	破綻スコア
システム: 笑顔でいたらいいことあるよ。<u-name>は思わず笑っちゃった話しとかない? ユーザ: あるよ システム: あるんだね。じゃあ、話そうか。 ユーザ: ゲーム実況者のキヨさんが本当に面白くてね	1 そうなんだ!ゲーム実況って、面白いよね!	0.081
	2 私もゲーム実況者さんのファンなんだ。面白いよね。	0.134
	3 へえ〜!そうなんだ。どんなゲームなの?気になるなあ。	0.132
	4 そうなんだ。その人が<u-name>の推し?	0.684
	5 へー!そうなんだ!それって有名な人?	0.817
システム: えへへ! その挨拶は、なんだか新鮮だね。 ユーザ: うふふ! システム: <u-name>って、よく笑うよね。明るい気分になる! ユーザ: そう?ありがとう	1 えへへ.....こちらこそ、ありがとう。	0.018
	2 こちらこそだよ。<u-name>は笑顔が素敵だね!	0.068
	3 こちらこそだよ。<u-name>は笑顔が素敵だね。	0.069
	4 こちらこそだよ、ありがとう!	0.136
	5 だって<u-name>が笑ってるって癒されるもん。	0.219
システム: ありがとう!<u-name>と一緒に食べようよ。今、何が1番食べたいの? ユーザ: カレー システム: カレーはおいしいよね。どのぐらいの辛さがいい? ユーザ: 中辛かな	1 いいね〜!お腹空いてきちゃったよ。	0.053
	2 おお〜、いいね、美味しそうだよ!今度一緒に食べに行こうよ。	0.054
	3 いいね!私も食べたいな〜。	0.097
	4 いいね。甘口?辛口?!辛?	0.781
	5 いいね。甘口?辛口?!?	0.829

節にて対話生成モデルの学習と、対話破綻検出モデルによる生成文の評価を行った。

4.1 対話破綻検出

本研究で構築したコーパスを学習データとし、応答文が破綻しているかどうか予測・評価した。

実験設定 表 4 に実験に用いたデータ数を示す。3 節のアノテーションで 5 人中 2 人以上が破綻と答えた文を正例、5 人中 5 人が自然と答えた文を負例とした。また、ランダムに 8 : 1 : 1 の比率でデータを分割し、学習: 検証: 評価データとした。

モデルは BERT [12] とし、東北大学が公開している事前学習済みモデル⁴⁾を利用した。fine-tuning では、応答文が破綻している場合に 1、自然な場合に 0 を出力するような回帰問題としモデルを学習した。また、検証データで精度が最大となったパラメータのモデルを評価に用いた。

実験結果 モデルによる対話破綻の予測精度を表 5 に示す。評価データに対する Accuracy は 0.633、ROC-AUC は 0.695 となった。

ワーカー間の破綻ラベルの一致数毎の、評価データに対するモデルの予測精度を表 6 に示す。5 人中 5 人が破綻と判定した事例に対する Accuracy は 0.703 であるのに対して 5 人中 2 人の場合は 0.580 であり、ワーカーの意見が揺れるほどモデルの予測精度も低い結果となった。破綻の判断は主観に頼っており正解ラベルにも揺れが含まれるため、モデルによる予測精度も低下していると考えられる。

4.2 応答文生成

構築した対話コーパスを学習データとし、応答文の生成実験を行った。

実験設定 藤原ら [13] による Transformer ベースの対話生成モデルを pretrained モデルとして利用し、本研究で構築したコーパスによって fine-tuning することで学習を行った。また、生成された応答文それぞれに 4.1 で構築した破綻検出モデルによる破綻スコアを付与した。なお、生成モデルの学習時には本コーパスを学習用、評価用に分割して利用した。

生成結果 表 7 に、評価データに対する応答文の生成例を示す。応答文をみると、日本語として違和感のある文章は少なく、ユーザーに対する回答として問題のない文章が生成できている。また 3 事例目に対する応答文 4,5 のように、ユーザーが回答しているにも関わらず再度質問をしているような文章には破綻スコアが高く推定されており、対話破綻を検出できている。一方で、1 事例目に対する応答文 4,5 のように、破綻スコアが高くなっているが必ずしも破綻とは言えない事例もあり、更なる分析が求められる。

5 おわりに

本研究では、実際の対話アプリログに対して対話破綻ラベルをアノテーションすることで、雑談対話コーパスを構築した。実験では、対話破綻の予測精度と応答文の生成結果を分析した。今後は対話システムの改善とともに、コーパスのデータ量追加や拡張を進めたい。

4) <https://github.com/cl-tohoku/bert-japanese>

参考文献

- [1] Adam S Miner, Liliana Laranjo, and A Baki Kocaballi. Chatbots in the fight against the covid-19 pandemic. *NPJ digital medicine*, Vol. 3, No. 1, pp. 1–4, 2020.
- [2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6000–6010, 2017.
- [3] Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.
- [4] Jason Williams, Antoine Raux, and Matthew Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, Vol. 7, No. 3, pp. 4–33, 2016.
- [5] Hannes Schulz, Jeremie Zumer, Layla El Asri, and Shikhar Kr Sarma. A frame tracking model for memory-enhanced dialogue systems. In *Proceedings of the 2nd Workshop on Representation Learning for NLP (ReplANLP)*, pp. 219–227, 2017.
- [6] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.
- [7] 藤村逸子, 大曾美恵子, 大島ディヴィッド義和. 会話コーパスの構築によるコミュニケーション研微. 藤村逸子, 滝沢直宏編『言語研究の技法: データの収集と分析』, pp. 43–72. ひつじ書房, 2019.
- [8] 東中竜一郎, 船越孝太郎, 荒木雅弘, 塚原裕史, 小林優佳, 水上雅博. テキストチャットを用いた雑談対話コーパスの構築と対話破綻の分析. *自然言語処理*, Vol. 23, No. 1, pp. 59–86, 2016.
- [9] 小磯花絵, 天谷晴香, 石本祐一, 居關友里子, 白田泰如, 柏野和佳子, 川端良子, 田中弥生, 伝康晴, 西川賢哉. 『日本語日常会話コーパス』 モニター公開版の設計と特徴. 言語処理学会第25回年次大会発表論文集, 2019.
- [10] 宇佐美まゆみ. BTSJ 日本語自然会話コーパス (トランスクリプト・音声) 2021年3月版. 機関拠点型基幹研究プロジェクト「日本語学習者のコミュニケーションの多角的解明. 国立国語研究所, 2021.
- [11] Deepali Aneja, Daniel McDuff, and Mary Czerwinski. Conversational error analysis in human-agent interaction. In *Proceedings of the 20th ACM international conference on intelligent virtual agents (IVA)*, pp. 1–8, 2020.
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pp. 4171–4186, 2019.
- [13] 藤原吏生, 岸波洋介, 今野颯人, 佐藤志貴, 佐藤汰亮, 宮脇峻平, 加藤拓真, 鈴木潤, 乾健太郎. Ilys aoba bot: 大規模ニューラル応答生成モデルとルールベースを統合した雑談対話システム. 第90回人工知能学会言語・音声理解と対話処理研究会 (第11回対話システムシンポジウム), 2020.