破綻類型情報に基づく雑談対話破綻検出

Breakdown detection based on taxonomy of errors in chat-oriented dialogue

堀井 朋　森 秀晃　林 卓矢　荒木 雅弘 *
Tomo Horii, Hideaki Mori, Takuya Hayashi, Masahiro Araki

京都工芸繊維大学
Kyoto Institute of Technology

Abstract: It is difficult to detect a breakdown of the chat dialogue arising from a variety of causes by one method. We examined breakdown detection methods that are suitable for each breakdown taxonomy that have been created by the dialog task group of the Project Next NLP. We also use a multilayer perceptron and the index output from the respective breakdown detector because we calculate the probability distribution of the breakdown detection result.

1　はじめに

雑談対話システムはユーザとの信頼感の醸成といった面で有用であるが，現在の対話システムではシステムがユーザの幅広い話題に対して適切な応答を出力し続けることは非常に困難である．そのような対話を破綻させる発話を出力前に検出することができれば，ユーザがシステムと対話を長く継続させることに繋がる．

しかし，雑談対話の破綻には様々な要因があり，一つの識別器で全ての破綻を検出することは困難であると考えられる．また，そこで，我々は破綻要因ごとに適切な手法を用いた識別器を構築し，それらを組み合わせることで適切に破綻を検出する破綻検出器を作成した．

2　提案手法

2.1　破綻類型

本研究では Project Next NLP の日本語対話タスクグループによる雑談対話の破綻原因類型化案 [2] に基づき，その類型毎に破綻識別器を作成し，それらを組み合わせる手法を提案する．

この類型化案では表 1 に示す通り，4 種の大分類を細分化した 16 種の小分類にまで類型化がなされている．しかし，本タスクに利用できる対話データ量では，小分類にまで細分化してしまうと各類型における破綻を学習できるデータ量が非常に少なくなってしまうため，充分に破綻を学習できないと考えた．そこで，今回は 4 つの大分類に対して有効と思われる検出手法を提案・検討した．

なお，大分類は対話のどの範囲に関連した破綻であるかという点を基準にして，以下のように分類されている．

発話 当該システム発話のみから破綻が認定できるケース

応答 直前のユーザ発話と当該システム発話から破綻が認定できるケース

文脈 対話開始時点から当該システム発話までの情報から破綻が認定できるケース

環境 破綻原因が，上記の 3 分類には当てはまらないケース

2.2　各類型に対する手法

2.2.1　発話類型

発話類型の破綻とは，基本的に発話の生成過程に問題がある場合—つまり構文的・意味的に日本語として妥当ではないものを指す。したがって，発話の単語間の繋がりが正しくないことが示されたならば，その発話を破綻として検出できるのではないかと考えた．

そこで，Google N-gram[3] を用いた単語間の共起確率を利用した手法で破綻を検出する．今回は 3-gram を用いて助詞の前後を考慮し，破綻を検出した．なお，形

*連絡先：京都工芸繊維大学工芸科学研究科
〒 606–8585 京都府左京区松ヶ崎御所海道町
E-mail: horii@ii.is.kit.ac.jp, mori@ii.is.kit.ac.jp
hayashi@ii.is.kit.ac.jp, araki@kit.ac.jp

表 1: Project Next NLP 対話タスク WG による破綻類型化

大分類	小分類	類型名	説明
発話	構文制約違反	構文的な誤り	日本語の文として正しくない
	意味制約違反	意味的な誤り	文としての意味が通らない
	不適切発話	解釈不能	著しく応答の体をなしていない
応答	量の公準違反	情報過不足	応答として内容の過不足がある
	質の公準違反	不理解	直前のユーザ発話を理解していない
	関係の公準違反	無関係	直前のユーザ発話の話題や意図と無関係
	様態の公準違反	意図不明	発話の意図が汲み取れない
	誤解	誤解	直前のユーザ発話の内容を誤って理解
文脈	量の公準違反	不要情報	冗長な発話
	質の公準違反	矛盾	発話内容や態度が急転換する発話
	関係の公準違反	無関係話題	話題が文脈から逸脱
	様態の公準違反	関連性不明	文脈のどの部分と関連しているか不明
	話題展開への不追随	不追随	話題展開後も前の話題を続ける
環境	無根拠	共通基盤欠如	根拠の無い主張
	矛盾	一般常識欠如	常識に反する主張
	非常識	社会性欠如	社会規範から外れ，相手を不快にする

態素解析には ipadic の拡張である NEologd[4] を辞書とした MeCab[1] を使用している．

2.2.2 応答類型

応答類型の破綻は，前発話との関係において協調の原則が守られていない破綻発話を指している．例えば，返答として情報が足りていなかったり，直前のユーザ発話を誤解していたり、といったものが挙げられる．つまり，破綻検出対象であるシステム発話とその直前のユーザ発話のみを素性として与えることで，応答類型に特化した破綻検出が行えるのではないかと考えられる．

そこで，破綻検出対象であるシステム発話とその直前のユーザ発話の発話対に対して，MeCab を用いて形態素解析を行い，単語の系列（単語の出現頻度に基づいた整数値のシーケンスとして符号化している）を Keras[2] の Embedding レイヤーを用いて 128 次元のベクトル系列に変換し，これを入力とする．

また，時系列データを扱うことから Long short-term memory[6] を中間層に用いた Recurrent Neural Network(LSTM-RNN) の一種である Bidirectional LSTM[7][8] を用いて破綻検出を行う．構造は図 1 の通りである．今回使用した Bidirectional LSTM のユニット数は 120(順方向と逆方向で 60 ずつ) とした．

LSTM とは RNN の勾配消失の問題を解決するために提案された時系列データに対するモデルの一種である．LSTM は記憶素子と入力ゲート・忘却ゲート・出力ゲートの 3 つのゲートから構成され，これらは情報をどの程度通すのかの制御に使われる．

Bidirectional LSTM とは，LSTM に「未来の入力から過去の出力を予測する」逆方向のモデルを考え，その出力を同一の出力層に統合というものである．つまり，時刻 $t-1$ の隠れ状態を t の隠れ状態の入力として用いる順方向の LSTM と時刻 $t+1$ の隠れ状態を t の隠れ状態の入力として用いる逆方向の LSTM を中間層に用いる．

2.2.3 文脈類型

対話の流れから判断できる不適切な発話・矛盾する情報の提供・不要な繰り返しなどがこの破綻類型に該当する．文脈類型の破綻を検出する際に必要となるのは，対話履歴に含まれる情報であると考えられる．しかし，対話開始時から当該システム発話までの全ての発話を素性として渡してしまうと，ある時点で破綻を引き起こしてしまったシステム発話の情報に引きずられ，それ以降のシステム発話も破綻であると検出してしまうことが考えられる．そのため，今回はある程度の範囲に限定して対話履歴を入力として扱うこととした．よって，入力には破綻検出対象のシステム発話からその前のユーザ発話・その前のシステム発話・更にもう一つ前のユーザ発話およびシステム発話の計 5 発話を単語に分割したものを使用した．

また，ネットワークとしては前後の文脈から単語ベ

[1] http://taku910.github.io/mecab/
[2] https://keras.io/ja/

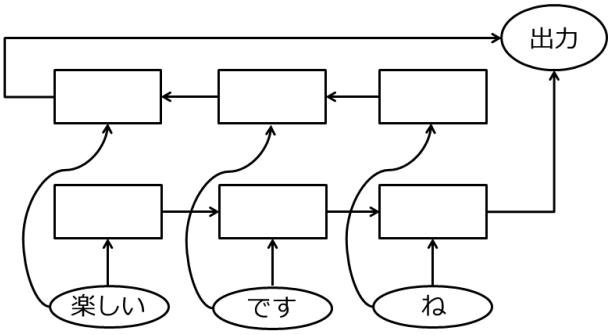

図 1: 応答類型で用いる BLSTM の模式図

クトルを計算できる Recurrent Convolutional Neural Networks(RCNN)[9] を用いて検出を行う．構造は図 2 に示す通りである．単語の出現順，その逆順の 2 つの方向から Bidirectional LSTM を用いて，それらを結合し単語ごとに一つのベクトルを生成し，全ての単語についてベクトル各要素の中の最大の値を取り出したベクトルを生成する．

2.2.4 環境類型

環境類型の破綻は，無根拠な主張や一般常識から乖離しているような発話や罵詈雑言といったようなものを指す．その中でも，一般常識から乖離しているような発話というものは「熱中症はいいですね」というようにネガティブな単語をポジティブな語で形容している発話が多い．今回はこのような破綻を検出することに焦点を当てた．提案した手法は以下の通りである．

まず，ネガティブな単語リストを人手で作成し，ワードネット [3] [5] を用いて単語リストの対義語および下位概念語を抽出した．なお，総数は 4,000 単語である．

そして，KNP[4] を利用して各システム発話に対してネガティブな単語にかかっている部分を抽出し，日本語評価極性辞書 [5] を用いて，ネガポジ判定を行う．ネガティブ単語にかかっている部分がポジティブであるならば，その発話は一般常識から乖離していると考え，破綻として検出する．

2.3 アノテーション割合の出力

本タスクではシステム発話に対する O・T・X のラベル一致率と分布距離とで評価する．そのため，出力された分類についての破綻率に基づいて，多層パーセプトロンを利用し，クラス数 N— 今回ならば各 O・T・X

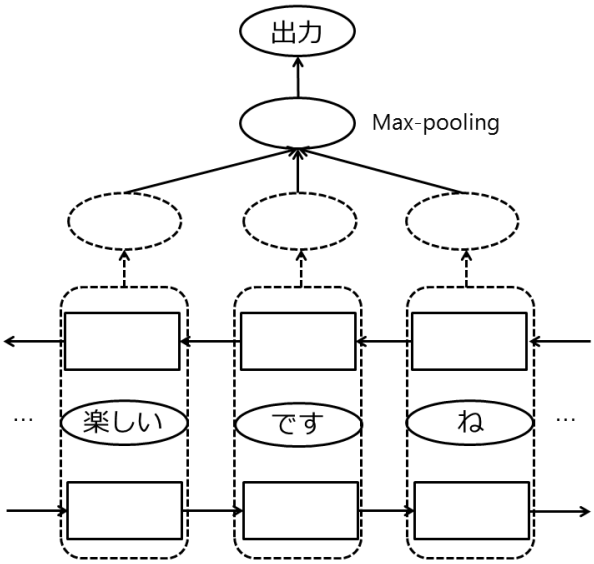

図 2: 文脈類型で用いる RCNN の模式図

に対するアノテーション割合 t を出力する．また，出力層の出力に関して softmax 関数を適用し，この出力 y に対して以下のような損失関数を用いた．

$$loss = -\sum_{i=0}^{N-1} log(y_i) * t_i \quad (1)$$

なお，今回使用した多層パーセプトロンは 3 層で，隠れ層のユニット数は 128 である．

3 評価実験

3.1 実験設定

提案した破綻検出手法の性能を評価するため，本タスク（破綻検出チャレンジ 2[1]）で配布されたデータを用いて，評価実験を行う．

配布されたデータのシステム発話には 30 名による O・T・X のアノテーションがなされているが，我々の提案手法では破綻発話についての破綻類型も必要となる．そのため，破綻発話に対して 3 名のアノテータにより破綻類型アノテーションをつけた．なお，このアノテーションは大分類についてのみのものであり，小分類についてのアノテーションは行っていない．

その際にどのシステム発話を破綻として扱うかを我々で決めなければならなかったが，昨年度の破綻検出チャレンジで使用したデータと基準をそろえるために，$t = 0.5$ として破綻発話を決定した．

[3] http://nlpwww.nict.go.jp/wn-ja/
[4] http://nlp.ist.i.kyoto-u.ac.jp/?KNP
[5] http://www.cl.ecei.tohoku.ac.jp/index.php?Open%20Reso-urces%2FJapanese%20Sentiment%20Polarity%20Dictionary

表 2: 評価結果（ラベル一致系統 , t=0.0）

		ACC	Pr(X)	Rc(X)	F(X)	Pr(T+X)	Rc(T+X)	F(T+X)
DCM	run1	0.476	0.423	0.663	0.516	0.768	0.599	0.673
	run2	0.455	0.453	0.573	0.506	0.753	0.825	0.787
	run3	0.462	0.500	0.309	0.382	0.810	0.604	0.692
DIT	run1	0.525	0.533	0.803	0.640	0.817	0.789	0.802
	run2	0.591	0.594	0.837	0.695	0.865	0.932	0.897
	run3	0.553	0.623	0.651	0.637	0.880	0.709	0.709
IRS	run1	0.520	0.492	0.762	0.598	0.743	0.745	0.744
	run2	0.498	0.506	0.688	0.583	0.750	0.857	0.800
	run3	0.455	0.505	0.442	0.471	0.758	0.647	0.694
Ave.	run1	0.507	0.483	0.743	0.585	0.776	0.711	0.740
	run2	0.515	0.518	0.699	0.595	0.789	0.871	0.828
	run3	0.490	0.543	0.467	0.497	0.816	0.653	0.698

表 3: 評価結果（分布距離系統 , t=0.0）

		JSD(O,T,X)	JSD(T+X)	JSD(O+T)	MSE(O,T,X)	MSE(T+X)	MSE(O+T)
DCM	run1	0.102	0.069	0.066	0.053	0.070	0.065
	run2	0.119	0.076	0.078	0.066	0.082	0.080
	run3	0.119	0.078	0.077	0.066	0.083	0.080
DIT	run1	0.076	0.054	0.052	0.043	0.055	0.060
	run2	0.064	0.042	0.042	0.036	0.043	0.050
	run3	0.066	0.043	0.045	0.037	0.043	0.053
IRS	run1	0.117	0.078	0.083	0.064	0.077	0.090
	run2	0.123	0.079	0.087	0.070	0.081	0.095
	run3	0.131	0.085	0.092	0.073	0.085	0.102
Ave.	run1	0.098	0.067	0.067	0.053	0.067	0.072
	run2	0.102	0.066	0.069	0.057	0.069	0.075
	run3	0.105	0.069	0.071	0.059	0.070	0.078

run1

破綻類型の大分類に基づいて，各々に対する破綻検出器を作成し，破綻検出を行う．また，破綻検出器から出力される破綻指数を元に多層パーセプトロンを用いて，各 O・T・X に対するアノテーション割合を算出した．

上述したように，発話類型には Google N-gram を用いた単語間の共起確率，応答類型には BLSTM(*epoch* 数は 200) を用いて，文脈類型には RCNN(*epoch* 数は 100，隠れ層のユニット数は 500) を用いて，環境類型には日本語のネガポジ判定を利用して破綻検出を行った．システム構造は図 3 に示す通りである．

応答類型と文脈類型の学習データは，開発データとして配布された DCM および IRS 各 50 対話の計 100 対話である．今回，DIT の対話データに関してはデータ内容を確認した結果，類型化が大いにブレるような発話（例 : 特定の分野に関して知識がないと本当かどうか分からない）がほとんどを占めており，本研究の前提である類型に沿った破綻であると見なせなかったため，学習の悪影響になる可能性を考慮し，学習データとしては使用していない．また，前処理として bot などを省くフィルタリング処理を行い収集した Twitter コーパス約 86 万ツイートの頻出単語に対して，上位 15 単語と 30000 位以下の単語は除く処理を行っている．

評価データは全ての run において DCM・IRS・DIT 各 50 対話の計 150 対話である．

run2

類型を用いずに，発話を単語に分割し RCNN を使い破綻検出を行う．昨年の破綻検出チャレンジのエラー分析 [10] から，出現割合が多かった応答・文脈の破綻

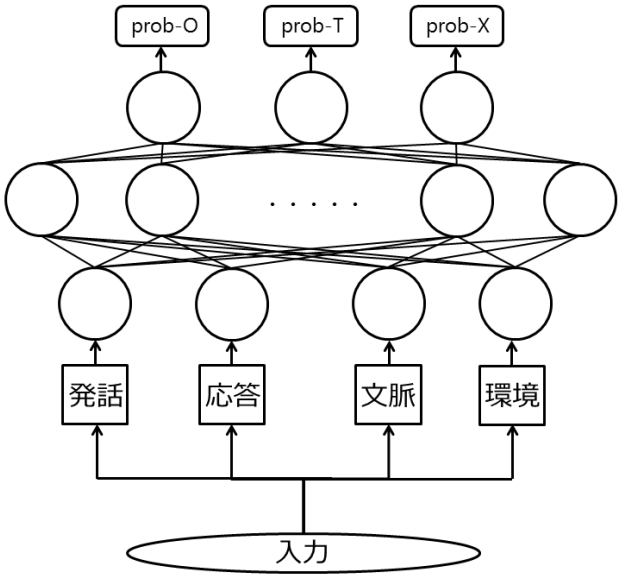

図 3: 提案システム構造

を上手く検出できるのではと考え，入力は当該システム発話およびその前のユーザ発話・もう一つ前のシステム発話としている．

なお，run2 においては学習データは開発データとして配布されたもの全て（150 対話）加え，昨年の破綻検出チャレンジで配布された 100 対話を用いており，run1 で行った前処理を加えた．また，epoch 数は 30，隠れ層のユニット数を 500 とした．

run3

run2 と同様に，発話を単語に分割し RCNN を用いて破綻検出を行った．学習データは run2 と同様であり前処理も加えているが，run3 では学習データ数の少なさ・過学習の危険性を考慮し epoch 数を 10 に，隠れ層のユニット数を 300 へと変更した．

3.2 結果

評価結果を表 2，表 3 に示す．表 2 についてはラベル一致系統に関する結果，表 3 には分布距離系統に関する結果についてまとめている．ラベル一致系統については run2 が，分布距離系統については run1 が全体的に優れた結果となっている．

特に run2 は DIT に対して最も良く働いているが，これは比較的破綻割合の大きい DIT データの影響により，DCM および IRS への推定精度が低下したためと推察できる．

また，run1 に関しては，多数のアノテータを用意した上で，DIT 対話データに対して的確な類型アノテーションを行い，学習データとして加えることで更なる性能向上を見込めると考えた．

4 おわりに

本稿では，破綻の類型化案に従って，破綻の検出に有用と思われる特徴や識別アルゴリズムを設定し，それらを組み合わせることで破綻検出を行った．

結果として，破綻類型を用いる用いないに関わらず同程度の性能を出せることが分かった．また，類型毎に識別器を作成する場合に関しては，各識別器およびアノテーション割合を推定するためのモジュールの最も精度の悪い部分に全体の性能が依存してしまう危険性があり，今回もそのような影響から，検出精度が劣る部分が生まれたと考えられる．

そのため，今後エラー分析を行い，改善すべき部分を明らかにした上で更なる精度向上を図る予定である．

参考文献

[1] 東中竜一郎，船越孝太郎，稲葉通将，荒瀬由紀，角森唯子: 対話破綻検出チャレンジ 2, 第 *78* 回言語・音声理解と対話処理研究会（第 *7* 回対話システムシンポジウム）(2016)

[2] 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.59–88 (2016)

[3] 工藤拓, 賀沢秀人: Google Ngram, *Web* 日本語 *N* グラム第 1 版, 言語資源協会発行

[4] Toshinori Sato: Neologism dictionary based on the language resources on the Web for Mecab, *https://github.com/neologd/mecab-ipadic-neologd*（2015）

[5] Francis Bond, Timothy Baldwin, Richard Fothergill, Kiyotaka Uchimoto: Japanese SemCor: A Sense-tagged Corpus of Japanese, *he 6th International Conference of the Global WordNet Association*, GWC–2012(2012)

[6] Sepp Hochreiter and Jurgen Schmidhube: Long short-term memory, *Neural* , Vol. 9,No. 8, pp.1735–1780（1997）

[7] Mike Schuster and Kuldip K Paliwal: Bidirectional recurrent neural networks. Signal Processing, *IEEE Transactions on*, Vol. 45, No. 11, pp.2673–2681（1997）

[8] 稲葉通将, 高橋健一: Long Short-Term Memory Recurrent Neural Network を用いた対話破綻検出, *SIG-SLUD*, SIG-SLUD 5.02, pp.57–60（2015）

[9] Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao: Recurrent Convolutional Neural Networks for Text Classification, *In AAAI*, pp. 2267–2273（2015）

[10] 堀井 朋, 荒木 雅弘: 類型毎の破綻検出器におけるエラー分析, 言語処理学会 第 *22* 回年次大会 発表論文集, pp. 437–440（2016）