人工知能学会研究会資料
SIG-SLUD-B502-11

RNN を用いた対話破綻検出器の構築

Construction of RNN-bsed Dialogue Breakdown Detector.

水上雅博 [1*]　杉山享志朗 [1]　Graham Neubig[1]　吉野幸一郎 [1]　Sakriani Sakti[1]　中村哲 [1]

[1] 奈良先端科学技術大学院大学，情報科学研究科

Abstract: We propose a dialogue breakdown detector based on the Recurrent Neural Netwoek (RNN). RNN is an extension of the neural network, which can consider old context information in sequential input. We construct a RNN network that receives utterance pairs of user and dialogue system as inputs, and outputs the breakdown label for each system's utterance. The input utterances are separated into vectors of words, cooccurrence words, and represented distributional sentence vectors. Finally, we combined these features and detector evaluated the breakdown detection accuracy.

1　はじめに

対話破綻検出は，雑談対話（非タスク指向型対話）システムと人間の対話ログから，対話の破綻（文脈上不適当なシステムの応答 [1, 2]）を検出するタスクである．対話システムの発話に対して，対話が破綻する可能性を事前に検出できれば，対話システムが対話破綻を起こす発話を行うことを避けることや，対話破綻後のエラーハンドリング戦略を準備することが可能となる．本研究は，この対話破綻検出の共有タスクである対話破綻検出チャレンジ [3] の一環として行う．

対話破綻検出のように，対話中に得られる情報から何らかの対話の状態を推定する研究として対話状態検出（Dialogue State Tracking; DST）があげられる．DST では，対話の状態を示す枠組みとしてユーザの意図を示すフレームとスロットが用意され，これを推定することで対話の状態推定を実現する．この DST の共通タスクである DST Challenge 2, DST Challenge 3 において，Recurrent Neural Netwoek (RNN) を用いた対話状態の推定を行う枠組み [4, 5] が提案され，非常に高い精度が報告されている．

本研究では，これらの研究を参考として，RNN を利用し，対話を系列的に取り扱う破綻検出器の構築に取り組む．

*連絡先：奈良先端科学技術大学院大学，情報科学研究科
〒 630-0192　奈良県生駒市高山町 8916-5
E-mail: masahiro-mi@is.naist.jp

2　対話破綻検出器の構築

2.1　RNN の概要

RNN はニューラルネットワークの一種であり，その特徴として，ネットワーク構造に前時刻までの状態を保持する隠れ層を持つ．前時刻までの状態を保持できる隠れ層は，過去の情報を記憶することで，現在の入力のみでなく過去の入力および出力を考慮した予測を可能とする．その結果，特に系列データに対して高い予測精度が期待される．

対話破綻検出において，RNN の隠れ層は対話の文脈や話題を記憶することができる．その結果，話題に沿っていない発話や，過去に同じ内容の発話など，その時点で入力される情報のみではわからない対話の破綻を検出することが期待できる．

2.2　RNN の拡張

先述の RNN の拡張として，Long-Short Term Memory (LSTM)[6] と呼ばれるニューラルネットワークが存在する．LSTM は，RNN と同様にネットワーク構造に前時刻の状態を保持する隠れ層（Memory Block; メモリブロック）と，隠れ層の各ノードの入力，出力，消去を制御する 3 種類のゲート（Input Gate, Output Gate, Forget Gate; 入力ゲート，出力ゲート，消去ゲート）を持つ．

LSTM は従来の RNN と同じく，過去の情報を記憶することで次の予測の精度を向上させることができる．加えて，ゲートによる隠れ層の制御によって，RNN と比較してより長距離の情報を隠れ層に残すことが可能となる．さらに，系列の変わり目等を学習し，RNN で

は困難であった系列の切れ目での記憶の消去を実現している．これによって，LSTM は RNN に比べて高い精度での破綻検出能力を持つことが期待できる．

2.3 入出力の設計

本研究では，対話破綻を検出するための素性として，これまでの対話破綻の分析 [7, 8] を考慮して，発話表層上の破綻，発話応答間の破綻，文意の破綻の三つの観点からそれぞれ効果的であると考えられる素性を提案する．それぞれの詳細を以下に示す．

単語頻度ベクトル（発話・応答） 発話表層上の特徴を捉える素性として，ユーザ発話およびシステム応答それぞれにおける単語の頻度をベクトル化して用いる．

発話・応答の共起単語頻度ベクトル 発話応答間における話題の遷移ややり取りを捉える素性として，ユーザ発話およびシステム応答で共起する単語の頻度をベクトル化して用いる．

doc2vec（発話・応答・共起単語） 出現する単語のスパース性を考慮し，文意を捉えるための素性として，単語頻度ベクトル，共起単語頻度ベクトルを圧縮して得られる単語分散表現を用いる．ベクトルの圧縮には doc2vec[9] を利用し，1024 次元のベクトルの単語分散表現を構築した．具体的には，Python ライブラリである gensim[1] を用いた．

これらの素性を対話のユーザ発話およびシステム応答ごとに抽出し，対話のターン経過に従って系列データとして破綻検出器に入力する．破綻検出器は，入力された素性を入力として，破綻の検出を行う．

対話破綻検出器は，出力として対話破綻ラベルの確率分布を出力する．破綻ラベルは X（破綻），T（おそらく破綻），O（破綻ではない）で表現され，それぞれのラベルに対する値の和が 1.0 になる．図 1 に，提案する対話破綻検出器の概要図を示す．

2.4 破綻検出器のチューニング

破綻検出器の学習は，対話破綻ラベルがアノテーションされた 2 種類の対話コーパス（Init， Rest）[1, 2] を用いて行い，チューニング用の対話コーパス（Dev）を用いてチューニングする．チューニングされた対話破綻検出器を用いて本評価用の対話コーパス（Eval）の破綻を検出する．具体的には，Init および Rest の計 1156 対話を用いて RNN の学習を 50Epoch 行い，各 Epoch 毎に Dev データに対する性能評価を行う [2]．

性能評価は，出力された対話破綻ラベルの割合うち，最大の値を持つ対話破綻ラベルと，実際にアノテーションされた対話破綻ラベルの一致率（チャレンジ用評価プログラムにおける閾値 $t = 0.0$ の場合の Accuracy に相当）を計算することで得られる．この一致率が最大となった対話破綻検出器を用いて Eval に対する破綻検出を行った．

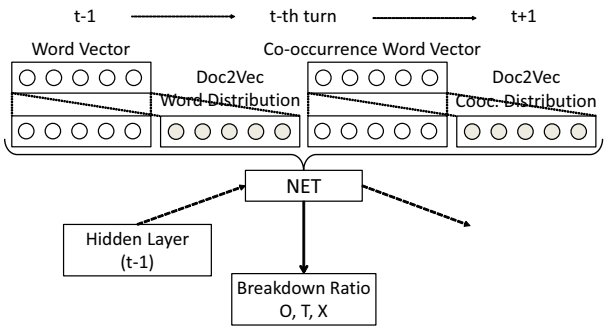

図 1: 対話破綻検出器の概要図

表 1: RNN を用いた対話破綻検出器の評価

	Score	Detail
Accuracy	0.47	(414/880)
Precision(X)	0.36	(110/307)
Recall(X)	0.44	(110/251)
F(X)	0.39	
Precision(X+T)	0.76	(241/318)
Recall(X+T)	0.44	(241/543)
F(X+T)	0.56	

3 評価実験

表 1 および表 2 に，RNN と LSTM を用いた対話破綻検出器の評価結果を，表 3 にベースライン検出器による評価結果 [3] を示す．評価指標は，チャレンジ用評価プログラムに従って，破綻ラベル全体の正解率である Accuracy，破綻ラベル X と破綻ラベル X+T に対する適合率である Precision,，再現率である Recall，Precision と Recall の調和平均である F を利用した．[4]．

これらの結果から，RNN および LSTM を用いた対話破綻検出器はベースライン検出器に比べてわずかに

[1] https://radimrehurek.com/gensim/

[2] LSTM を用いた対話破綻検出器の学習は，時間の都合上，10Epoch のみ行った．

[3] ベースライン検出器の学習には Init, Rest, Dev を混合して利用した．

[4] 評価にはそれぞれチャレンジ用評価プログラムにおける閾値 $t = 0.1$ の場合を用いた．

表 2: LSTM を用いた対話破綻検出器の評価

	Score	Detail
Accuracy	0.47	(414/880)
Precision(X)	0.57	(4/7)
Recall(X)	0.02	(4/251)
F(X)	0.03	
Precision(X+T)	0.86	(6/7)
Recall(X+T)	0.01	(6/543)
F(X+T)	0.02	

表 3: ベースライン対話破綻検出器の評価

	Score	Detail
Accuracy	0.46	(405/880)
Precision(X)	0.43	(42/97)
Recall(X)	0.17	(42/251)
F(X)	0.24	
Precision(X+T)	0.76	(356/469)
Recall(X+T)	0.66	(356/543)
F(X+T)	0.70	

Accuracy を改善した．また，RNN を用いた対話破綻検出器は，破綻ラベル X に対する F 値についてもベースラインに比べて改善する傾向があった．

3.1 傾向と分析

RNN を用いた対話破綻検出器の出力傾向を分析するため，検出器の出力ラベルとアノテーションされた破綻ラベルとの混同行列を表 4 に示す．

混同行列から，RNN を用いた対話破綻検出器の出力はおおむねアノテーションされている破綻ラベルと同じものが最大となっている．一方で，T（おそらく破綻）の出力が著しく少なく，ほとんどが X（破綻）または O（破綻でない）として出力されている．これらのことから，RNN を用いた対話破綻検出器は T（おそらく破綻）を X（破綻），O（破綻でない）と検出してしまう傾向があると考えられる．

次に，RNN を用いた対話破綻検出器によって検出が改善した例について表 5 に示す．一つ目の例では，ユーザ発話とシステム応答に共通して「サラリーマン」という単語が入っており，一見話題を継続して応答を行っているように見えるが，文意が不明であり応答として不適切である．提案した破綻検出器では，doc2vec を用いた単語分散表現を用いて文意を捉える素性を用いているため，ベースライン検出器では検出できなかった破綻ラベル X が検出可能となったと考えられる．二

表 4: RNN を用いた対話破綻検出器における付与されたラベルと正解ラベルの混同行列

	Predicted			
	X	T	O	
X	**110**	3	138	251
Annotated T	90	**4**	124	218
O	107	4	**300**	411
	307	11	562	880

つ目の例では，ユーザの質問に対して，システムが答えているように見えるが,「面白いか」という問いに対して「時間が足りてない」は正しい応答ではない．提案した破綻検出器では，質問に対して応答を行うようなやり取りを捉える素性として共起単語頻度ベクトルを用いているため，質問に対する応答の破綻を検出できたと考えられる．一方で，三つ目の例では，破綻の種類としては 2 番目に近いと考えられるが，人間が明らかに破綻であるとわかる発話対に対して破綻ラベル X の検出が失敗している．

次に，LSTM を用いた対話破綻検出器の出力した破綻ラベルは O:873， T:0， X:7 となり，ほぼ O ラベルしか出力していなかった．この問題の理由として，LSTM のネットワークの学習不足が考えられる．RNN が一つの素性に対してノードの重みのパラメタを 1 つ学習するのに対して，LSTM は一つの素性に対してノードの重みと 3 つの制御ゲートの重みの計 4 つのパラメタを学習する．そのため，RNN の学習量（6 時間/50Epoch）に比べて多大な時間が必要となり，今回の学習量（6 時間/10Epoch）では十分な学習が行われなかったと考えられる．

4 まとめ

本研究では，対話破綻検出を行う手法として，系列データに対して高い識別能力を持つ RNN と，その拡張である LSTM を用いた対話破綻検出器を提案し，その精度を評価した．評価の結果，RNN および LSTM を用いた対話破綻検出器はベースラインに比べて精度の改善が確認できた．

今後の課題として，隠れ層のノード数を変化させた場合の精度の変化と，学習量と破綻検出の精度の関係の評価を行う．

参考文献

[1] 東中竜一郎, 船越孝太郎. Project Next NLP 対話タスクにおける雑談対話データの収集と対話破綻アノ

表 5: RNN とベースラインとの検出結果の比較

User Utterance	System Response	Annotation	RNN	Baseline
サラリーマンなので、 お昼休みに見ることが多いですよ。	片手間でサラリーマンがついてます？	X	X	O
小説は面白いですか？	時間が足りてないかも	X	X	O
何が食べたいですか？	あちらこちらで イルミネーションがキレイですね	X	O	O
退屈してたのですか？	休みの日は良く出掛けるんですか？	T	T	X

テーション (第 5 回対話システムシンポジウムおよび一般). 言語・音声理解と対話処理研究会, Vol. 72, pp. 45–50, dec 2014.

[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. Project Next NLP 対話タスク : 雑談対話データの収集と対話破綻アノテーションおよびその類型化.

[3] 東中竜一郎, 船越孝太郎, 小林優佳, 稲葉通将. 対話破綻検出チャレンジ, 2015.

[4] Matthew Henderson, Blaise Thomson, and Steve Young. Word-based dialog state tracking with recurrent neural networks. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 292, 2014.

[5] Matthew Henderson, Blaise Thomson, and Steve Young. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pp. 360–365. IEEE, 2014.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[7] Ryuichiro Higashinaka, Kotaro Funakoshi, Masahiro Araki, Hiroshi Tsukahara, Yuka Kobayashi, and Masahiro Mizukami. Towards taxonomy of errors in chat-oriented dialogue systems. In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 87, 2015.

[8] Ryuichiro Higashinaka, Masahiro Mizukami, Kotaro Funakoshi, Masahiro Araki, Hiroshi Tsukahara, and Yuka Kobayashi. Fatal or not? finding errors that lead to dialogue breakdowns in chat-oriented dialogue systems. In *The 2015 Conference on Empirical Methods on Natural Language Processing.*, pp. 2243–2248, September 2015.

[9] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *Proceedings of the 31st International Conference on Machine Learning*, 2014.