料理アシスト対話システムにおけるユーザ発話のクラス分類*

☆山田燿司, 滝口哲也, 有木康雄 (神戸大)

1 はじめに

本研究では, 料理に取り組むユーザを補助する“ 料理アシスト対話システム ”の構築を目的とした, ユーザ発話のクラス分類を行う. この対話システムは, 料理の手順や材料の分量といった料理に関する発話 (質問) に対して応答する. システムは発話の種類毎に, 手順データベース（DB）や材料 DB 等の異なる DB を参照する. この点から, より適切な応答を生成するために, ユーザ発話のクラス分類を行い, 参照すべき DB を特定する. このクラス分類について, RNN Encoder-Decoder 等の手法を用いて比較実験を行う.

2 料理アシスト対話システム

2.1 料理アシスト対話システムの構成

近年, 料理の補助を目的とする対話システムについての研究が重視されている [1]. 例えば, Fig.1 のように, システムは料理の手順や材料の分量等, 料理に関する質問に対して応答する. 料理中, ユーザは食材や調理器具を手に持つため, このような音声でのシステム操作が求められる.

この対話システムは, 手順 DB や材料 DB 等の複数の DB を参照する. 各 DB の詳細を Fig. 2 に記載する. 各 DB は模擬対話から得たコーパスや楽天レシピを用いて構築される. システムは,“ 手順に関する質問 ”,“ 材料に関する質問 ”や“ 雑談 ”等, ユーザ発話の種類毎に異なる DB を参照し, 応答を生成する. 従って, より適切な応答を生成するために, ユーザ発話がどの DB に対応する発話であるかを分類する必要がある. このような発話の分類は, 「DSTC (Dialogue State Tracking Challenge)」において, 対話の内容を解析することと同様に, より正確な対話の実現のために重要である [2]. このユーザ発話のクラス分類を行うために, 模擬対話から得たコーパスに含まれるユーザ発話を, 学習データとテストデータに分割し, ラベル付を行い, 分類手法の比較実験を行った. そして, 最も良い結果が得られた分類手法を用いてシステムを構築する.

システムの構成を Fig.3 に記載する. 初めに, ユーザの発話を発話解析部でベクトルに変換する. 次に, 得られた発話ベクトルに対して, 模擬対話のユーザ発話を用いてクラス分類部で発話クラスを特定する. そして, ユーザ発話や発話クラス, 各種 DB を用いて, 応答生成部にてシステム発話を出力する.

Usr : カレーを作りたい.
Sys : では, こちらのレシピはいかがでしょうか?
Usr : いいね. じゃあまず何をしたらいい?
Sys : まず、玉ねぎをくし切りにします.
Usr : 玉ねぎは何個使う?
Sys : 玉ねぎは1個使います.
Usr : 玉ねぎのくし切りの仕方は?
Sys : 玉ねぎのくし切りで検索します.

Fig. 1 対話例

2.2 学習データ

楽天データセット (提供 : 楽天株式会社, 国立情報学研究所) 内の楽天レシピには約 80 万のレシピが存在し, 各レシピは料理手順に関するデータや材料に関するデータで構成されている. 模擬対話では, ユーザ役とシステム役に分かれ, 楽天レシピを基に料理に取り組みながら対話を行う. ユーザ役は, 実際に料理を行い, 料理の過程で分からないことがあればシステム役に質問する. システム役は, ユーザ役の質問に対して, 楽天レシピのデータや web 検索を基に応答する. 今回, 楽天レシピに含まれるレシピの中から 20 品目を選び, 1 名の料理経験者であるシステム役と, 10 名の料理経験者もしくは未経験者であるユーザ役で模擬対話を 20 回行い, 1218 文のシステム発話と 1218 文のユーザ発話を得た. システム発話は応答を生成する DB の構築に用いられ, ユーザ発話はクラス分類の学習データとテストデータとして用いられる. ラベル付は, 各ユーザ発話に対して, Fig.2 の 8 種の DB の中のどの DB で応答可能であるかを, ラベル付与経験者 3 名が判別し, 1 から 8 のラベルを付与した.

3 ユーザ発話のクラス分類

ラベル付されたユーザ発話に対してクラス分類を行う. 初めに, Bag-of-Words (BoW) もしくは Recurrent Neural Network (RNN) Encoder-Decoder を用いて, ユーザ発話をベクトルに変換する. その後, 発話ベクトルを Deep Neural Network (DNN) もしくは Support Vector Machine (SVM) を用いて分類する.

*Classification of user utterances in the cooking assist dialog system. by YAMADA, Yôji, TAKIGUCHI, Tetsuya, ARIKI, Yasuo (Kobe University)

1. 手順DB（楽天レシピ+模擬対話）
   料理の手順に関する質問に応答
   Usr :「何をしたらいい?」
   Sys :「まず、玉ねぎをくし切りにします.」
2. 材料DB（楽天レシピ+模擬対話）
   材料に関する質問に応答
   Usr :「玉ねぎは何個使う?」
   Sys :「玉ねぎは2個使います.」
3. 共通知識DB（Web検索+模擬対話）
   どの料理にも共通する質問に応答
   Usr :「玉ねぎのくし切りの仕方は?」
   Sys :「玉ねぎのくし切りで検索します.」
4. 特定知識DB（模擬対話）
   特定の料理にのみ適用される質問に応答
   Usr :「カレーにりんご足してもいい?」
   Sys :「すりおろして入れましょう」
5. 雑談DB（雑談コーパス+模擬対話）
   雑談に応答
   Usr :「お腹が空いた」
   Sys :「空腹の時ほど美味しく感じますよね.」
6. 機能DB（模擬対話）
   システム機能の操作に関する要求に応答
   Usr :「3分経ったら教えて」
   Sys :「タイマーを3分に設定します」
7. レシピDB（楽天レシピ）
   レシピの検索に応答
   Usr :「カレーを作りたい」
   Sys :「こちらのレシピはいかがでしょうか?」
8. その他（模擬対話）
   相槌や同じ発話の繰り返しに対してルールベースに応答
   Usr :「分かりました.」
   Usr :「もう一度言って下さい.」

Fig. 2　DB 一覧

3.1　ユーザ発話のベクトル化

3.1.1　Bag-of-Words

BoW は, テキスト化した発話を形態素解析を用いて単語に分解し, 各単語の頻出回数を並べることで, ベクトルを生成する [3]. 今回得られた 1218 文のユーザ発話には, 1094 種の単語が含まれており, それらの頻出回数を基に 1094 次元のベクトルに変換できる. しかし, 頻出回数が極端に少ない単語は, 有効な発話の特徴量にならず, クラス分類の妨げとなる. この様な単語を取り除くことで, 低次元且つ有効な特徴量を持つ発話ベクトルが得られる.

3.1.2　RNN Encoder-Decoder

RNN Encoder-Decoder を用いた発話のベクトル変換を Fig. 4 に示す [4, 5, 6]. x_1 から x_T は, ユーザ発話 1 文に含まれる各単語に対応した one-hot のベクトルであり, T は発話内の単語数を表す. Encoder 部の RNN では, 入力系列 $x_{1:T}$ を順に受け取り, 式 (1) に従って隠れ層 $h_e^{<t>}$ を更新し, $h_e^{<T>}$ を Decoder 部の入力とする. Decoder 部の RNN では, $h_e^{<T>}$ を $h_d^{<1>}$

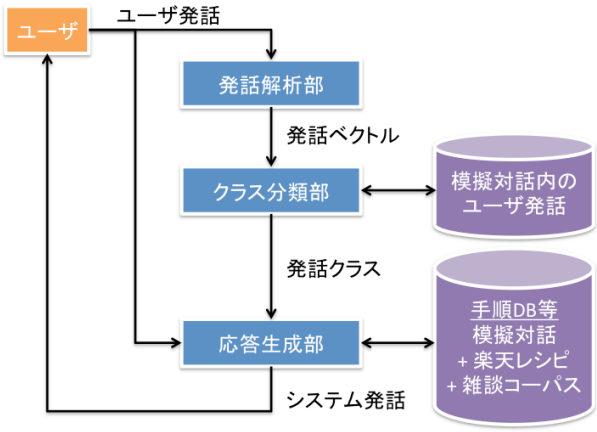

Fig. 3　システム構成

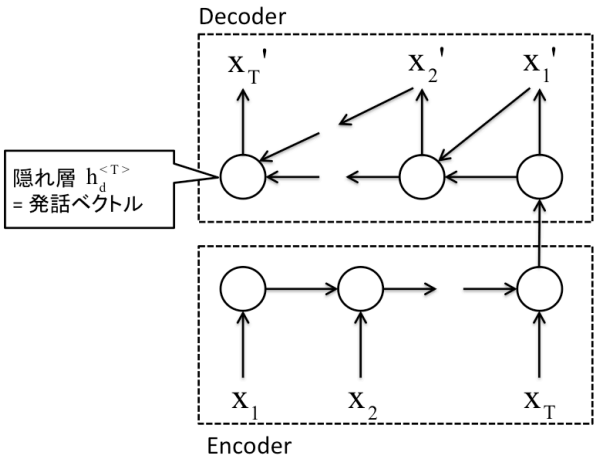

Fig. 4　RNN Encoder-Decoder を用いた発話のベクトル変換

の入力として受け取り, $t > 1$ において式 (2) に従って隠れ層 $h_d^{<t>}$ を更新し, $x'_{1:T}$ を出力する.

$$h_e^{<t>} = LSTM(h_e^{<t-1>}, x_t) \tag{1}$$

$$h_d^{<t>} = LSTM(h_d^{<t-1>}, x'_{t-1}) \tag{2}$$

隠れ層における活性化関数には Long Short Term Memory (LSTM) を用いる. 式 (3) の Mean-Square-Error (MSE) [1]を用いて, Encoder 部の入力 $x_{1:T}$ と Decoder 部の出力 $x'_{1:T}$ の誤差を求め, 誤差逆伝播より各ユニット間のパラメータを更新し, $x'_{1:T}$ が $x_{1:T}$ に近づくように学習を行う. このパラメータ学習をユーザ発話 1218 文より行い, 学習後の RNN Encoder-Decoder に再びユーザ発話を 1 文ずつ入力して得られる, 各 $h_d^{<T>}$ を発話ベクトルとする.

$$MSE = \frac{1}{T}\sum_{t=1}^{T} ||x_t - x'_t||^2 \tag{3}$$

[1] $||\cdot||^2$ は L2 ノルムを表す.

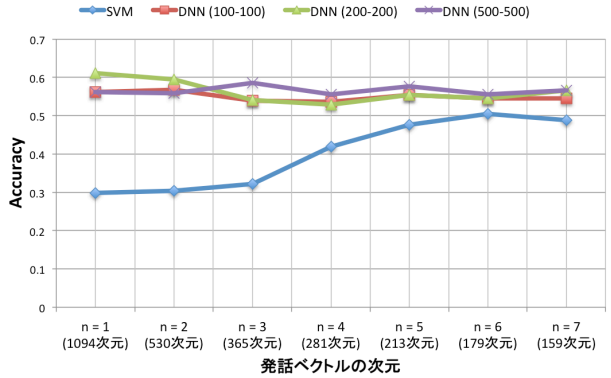

Fig. 5　BoW を用いたクラス分類の結果

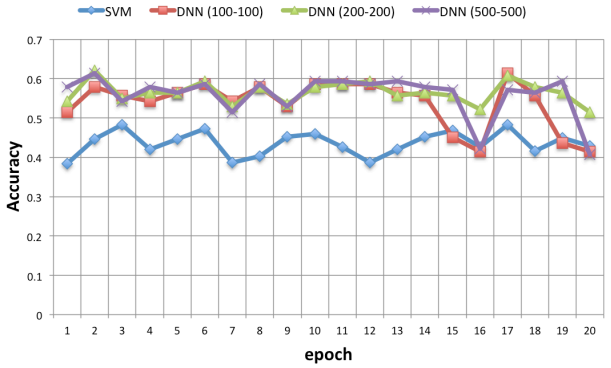

Fig. 6　RNN Encoder-Decoder を用いたクラス分類

このように, Encoder 部の入力にユーザ発話を設定し, Decoder 部の出力が Encoder 部の入力に近づくようにパラメータの学習を行い, そのパラメータを用いて, 再び順伝播をすることで得られる隠れ層の値 $h_d^{<T>}$ を発話ベクトルとする.

3.2　クラス分類

3.1 より得られた発話ベクトルと, 各発話ベクトルに付与された 1 から 8 のラベルを基にクラス分類を行う. SVM では one-vs.-rest を適用する. 特定の 1 クラスに属するか, 他の 7 クラスのいずれかに属するかを識別する 2 クラス分類器を, 学習データから 8 個構築し, テストデータのラベルを推定する. DNN では入力を発話ベクトル, 出力をラベルに設定し, 学習データを用いてパラメータを学習し, テストデータのラベルを推定する. 活性化関数にはソフトマックス関数, 誤差関数には交差エントロピーを用いる [7].

また, 形態素に基づくクラス分類も行う [8]. まず, 各発話データに対して形態素解析[2]を行う. テストデータの発話を S_i, 学習データの発話を S_j とすると, S_i のラベル L_i は, 式 (4) を用いて, S_i と最も類似する S_j に付与されたラベルと同一とする.

$$L_i = L_{\hat{j}},\ \hat{j} = \arg\max_j \frac{2M_{ij}}{M_i + M_j} \qquad (4)$$

M_i, M_j はそれぞれ S_i, S_j の形態素数, M_{ij} は S_i と S_j で一致する形態素の数を表す.

[2]MeCab http://taku910.github.io/mecab/

4　比較実験

クラス分類により推定されたラベルとラベル付与経験者が付与したラベルを比較し, 分類器の精度を測定する. BoW もしくは RNN Encoder-Decoder によるベクトル変換と, SVM もしくは DNN によるクラス分類を組み合わせた 4 種の手法. また, 形態素に基づく手法を加えた, 計 5 種の分類器について比較する.

4.1　実験設定

BoW は, 頻出回数が n 回未満の単語を取り除き, n を 1 から 7 に設定することで, 1094 次元から 159 次元の間の発話ベクトルを生成する [10]. RNN Encoder-Decoder は, epoch を 20, Encoder 部と Decoder 部の隠れ層のユニット数を共に 500 に設定し, 各 epoch 毎に 500 次元の発話ベクトルを生成する. SVM のハイパーパラメータはグリッドサーチを基に選択する. 今回, 分類精度が最も高くなる様に, コストパラメータ C=1, RBF カーネルのパラメータ: γ =0.001 に設定した. DNN は, 入力層-隠れ層-隠れ層-出力層で構成され, 2 層の隠れ層のユニット数を 100-100, 200-200, 500-500, epoch を 20 に設定し, 学習を行う. 実験結果では, 20 回の学習の中で, 最も正解率が高い結果を表示する. また, Cross-Validation を基に, ユーザ発話 1218 文を, 学習データ 1097 文とテストデータ 121 文に分割し, 学習とテストを行う.

4.2　実験結果

Fig. 5 は BoW から発話ベクトルを生成し, SVM と DNN を用いてクラス分類を行った結果を示す. SVM は発話ベクトルの次元数が少ないほど正解率が上昇し, DNN は次元数と正解率の相関性が低いことが分かった. Fig. 6 は RNN Encoder-Decoder から発話ベクトルを生成し, SVM と DNN を用いてクラス分類を行った結果を示す. epoch 2 で生成されたベクトルに対し, DNN の隠れ層を 200-200 に設定し, 分類を行った場合に, 最も高い正解率を得た.

Table 1 は, 形態素に基づくクラス分類も含め, 各クラス分類の結果の中で, 最も正解率が高い結果を示す. SVM より DNN の方が分類精度が高く, RNN Encoder-Decoder と DNN を組み合わせた手法が最も有効であった. しかし, 最高正解率が 62.14% と, 他のクラス分類 [9, 10] と比較して低い正解率を示した. その原因としては, ユーザの発話を制限しなかったために, 発話内容が多様であったこと, また, 収集したユーザ発話のデータが 1218 文と少量であり, 学習が不十分であったことが考えられる. この点から, さらにデータを収集し, より大きなデータを用いて分類器の検証を行う必要がある.

Table 1　比較実験結果

	正解率（%）
形態素	51.23
BoW + SVM	50.50
BoW + DNN	61.15
RNN Encoder-Decoder + SVM	48.19
RNN Encoder-Decoder + DNN	62.14

5　おわりに

本研究では, 料理アシスト対話システムの構築に必要である, ユーザ発話のクラス分類について, いくつかの手法を用いて比較実験を行った. 実験結果より, RNN Encoder-Decoder を用いて発話ベクトルを生成し, 発話ベクトルに対して DNN を用いて分類を行う手法が最も有効であることが分かった. 今後の課題として, 今回の結果を基にユーザ発話の分類を行い, 分類結果に応じて適切な DB を参照し, 応答を生成する対話システムを構築する. そして, 対話システムの応答の精度を測定する予定である.

参考文献

[1] Joana Paulo Pardal and Nuno J. Mamede, ”Starting to Cook a Coaching Dialogue System in the Olympus framework” Proceeding of the Paralinguistic Information and its Integration in Spoken Dialouge Systems Workshop, 255-267, 2011.

[2] Hongjie Shi, Takashi Ushio, Mitsuru Endo, Katsuyoshi Yamagami, and Noriaki Horii, ”Convolutional Neural Networks for Multi-topic Dialog State Tracking” IWSDS (International Workshop on Spoken Dialogue System) 2016.

[3] 真嶋 温佳, 藤田 洋子, トーレス ラファエル, 川波 弘道, 原 直, 松井 知子, 猿渡 洋, 鹿野 清宏, ”音声情報案内システムにおける Bag-of-Words を用いた無効入力の棄却” 情報処理学会論文誌, Vol.54 No.2 443-451 2013

[4] Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen and Steve Young, ”On-line Active Reward Learning for Policy Optimisation in Spoken Dialogue Systems” ACL, 2016.

[5] Kyunghyun Cho, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshuna Bengio, ”Learning Phrase Representation using RNN Encoder-Decoder for Statistical Machine Translation” EMNLP, 2014.

[6] ”Chainer と RNN と機械翻訳”, http://qiita.com/odashi_t/items/a1be7c4964fbea6a116e

[7] 岡谷貴之, ”深層学習” 講談社 2014

[8] 入江 友紀, 松原 茂樹, 河口 信夫, 山口 由紀子, 稲垣 康善, ”意図タグ付きコーパスを用いた発話意図推定手法” 言語・音声理解と対話処理研究会 38, 7-12, 2003.

[9] 岩下 薫, 嶋田 和考, 遠藤 勉, ”対話システムにおけるユーザの発話意図推定”

[10] 加藤 玲大, 馬 青, 村田 真樹, ”深層学習を用いた QA サイト質問文のカテゴリ分類” 情報処理学会研究報告, No.10, 2016