6T-7

ユーザの言い淀みによる発話の誤分割を事後的に回復する音声対話システム

堀田 尚希 † 駒谷 和範 ‡ 佐藤 理史 ‡

† 名古屋大学 工学部電気電子・情報工学科 ‡ 名古屋大学大学院 工学研究科 電子情報システム専攻

1. はじめに

音声対話システムにおいて，音声認識誤りの原因のひとつに発話区間検出誤りがある．一般的な発話区間検出法では，音声信号の振幅や零交差数により無音区間を判定し，その無音区間がある決まった閾値より長いときに，ユーザの発話が終了したとみなす．一方で，ユーザが長い単語や文を発話するときに言い淀む場合がある．この言い淀みが長い場合，発話区間が誤って分割される可能性がある．この例を図1に示す．ユーザが「ナゴヤドーム前矢田駅に行きたい」と発話したときに,「ナゴヤドーム前」と「矢田駅に行きたい」の間に，言い淀みが発生した例である．ここで「ナゴヤドーム前矢田」は，名古屋市営地下鉄に存在する駅の名前である．このように発話区間が誤分割された場合，システムはユーザ発話の一部分に対する音声認識結果に基づき，応答することになる．その応答は，多くの場合適切ではない．この例では，誤分割された後半部分の発話「矢田駅に行きたい」の音声認識結果に基づいて応答が行われる．具体的には「(大阪府の) 矢田までの行き方を表示します」のような応答であり，適切ではない．

本研究では，発話が誤分割された場合に，誤分割された発話を結合して再度音声認識を行うことで，発話区間検出誤りを事後的に修復する．これは，ユーザに対して可能な限り素早い応答をするシステムには必須の機能である．具体的には，

1. ユーザの発話終了後，できるだけ早く応答を開始する
2. システムの発話中にユーザが話し始めた場合，システムは発話を中断し，そのユーザ発話の音声認識を行う (バージイン機能)

というシステムである．このようなシステムでは，ユーザが言い淀んだ場合に，その短い無音区間によってユーザ発話が誤って分割され，システムが発話を開始してしまうという問題が起こる．

発話を結合する具体例を図2に示す．この例では「ナゴヤドーム前」と「矢田駅に行きたい」のように誤分割された発話を結合して,「ナゴヤドーム前矢田駅に行きたい」という1つの発話として再度音声認識を行っている．これにより，できるだけ早く応答を開始するシステムにおいても，事後的に正しい発話区間に対する音声認識が可能となる．

2. 誤分割された発話を結合するシステム

誤分割された発話を結合して再度音声認識を行うシステムを実装するには，以下の5つの処理が必要である．ここで誤って分割された発話の前半を1発話目，後半を2発話目とする．

Spoken Dialogue System that Retroactively Recovers Incorrectly-Divided Utterances Caused by User's Disfluency: *Naoki Hotta, Kazunori Komatani, and Satoshi Sato (Nagoya Univ.)*

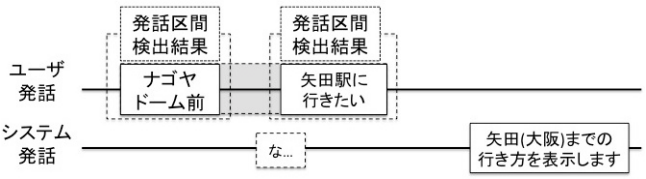

図 1: 言い淀みに対する発話の誤分割

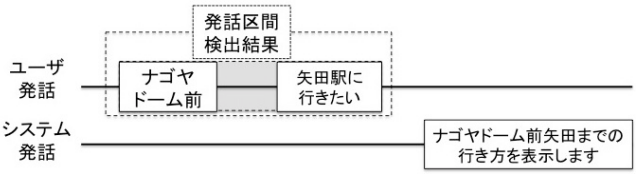

図 2: 言い淀みに対する発話の結合

1. 発話を結合すべきかどうかを判定する
2. 1発話目の音声認識結果に対するシステムの応答発話を停止する
3. 2発話目の音声認識結果を捨てる
4. 録音しておいた音声ファイルを結合し，それに対して認識を行う
5. 音声ファイルを結合し，認識している間に，フィラーを生成する．

まず，連続する2つの発話が本来1発話であった場合に，2つの発話を結合する一連の処理を行う．この判定は，現時点では，単純に2発話間の時間的間隔により行っている．本来は1発話であった場合，誤分割された2つの発話断片に対して，システムは応答しようとしてしまうため，これらを破棄する必要がある．2と3の処理はこれらに対応する．4は，2つの発話を結合し再度音声認識を行う処理である．現在の実装ではこの処理に時間がかかるため，不自然な間が発生するのを防ぐために，5の処理が必要となる．

このシステムを，MMDAgent[1] のプラグインとして実装する．我々はユーザの発話に対し素早く応答ができる音声対話システムを目標としており，MMDAgent はこの条件を満たす．MMDAgent は音声認識や音声合成など全ての機能がプラグインとして実装されている．各プラグインは Global Message Queue と呼ばれるキューからメッセージを受け取り，各プラグインでの処理結果を，メッセージとしてキューに送る．我々が実装したプラグインも，MMDAgent の他の部分を変更することなく，追加可能である．

2.1 処理の概要

新たに実装するシステムの状態遷移図を図3に示す．図3の四角は状態を，矢印は状態遷移を示している．また，矢印上の文字は，状態遷移が起こる条件を示す．灰

色で示したものが，新たにプラグインとして加えた遷移である．ここで2つの状態を定義する．まず，システムがユーザの発話を待っている状態と，ユーザが発話している状態を合わせて「待機状態」とする．さらに，システムが発話をしている状態を「システム発話中」とする．我々のプラグイン実装前のシステムでは，(1)「待機状態」でユーザの応答を待つ，(2) ユーザ発話に対する音声認識結果が得られたら「システム発話中状態」に遷移し，システムが発話を行う，(3) システムの発話が終了したら，再び「待機状態」に戻る，というような状態遷移である．

これに対して本稿ではまずシステムの発話を停止するために,「バージイン状態」を追加する．これは，システム発話中にユーザが話し始めた状態である．システム発話中にユーザ発話の開始が検出されると,「バージイン状態」に遷移し，システムの発話を停止した後「待機状態」に遷移する．これによりユーザ発話中にシステムが話し始めた場合でも，システムの発話を停止させることができる．

次に発話を連結するために,「音声結合・認識状態」を追加する．これは，発話を結合し再度音声認識を行う動作を行っている状態である．図4は，誤分割された発話を結合する際の処理の流れを表している．追加するプラグインでは，現在のユーザ発話の録音とともに発話の開始と終了時刻を記録している．本稿では，もし現在のユーザ発話の終了時刻と次のユーザ発話の開始時刻 (以下，発話の間) が一定時間以内であれば，その発話は本来1発話であったとみなし,「発話結合開始」を表すメッセージを，MMDAgent のキューに送る．その後，2つのユーザ発話を結合し，これに対して音声認識を行う．音声認識が終了したら「音声認識終了」を表すメッセージと音声認識結果を，キューに送る．音声認識結果をキューに送ることで，この音声認識結果に基づきユーザへの応答を生成する．現状の実装では，音声認識の際に結合した音声ファイルの長さ分 (1秒から2秒程度) の遅延が発生する．これは2つの音声ファイルが完全に得られた後に結合しているためである．この間をつなぐために「えっと...」というフィラーを入れることで，対話に不自然な間が生じないようにしている．

図3の状態遷移図では,「発話結合開始」メッセージを受け取ると，システム発話を停止してから「音声結合・認識状態」に遷移する．このとき，システム発話を停止することで，結合する際に2発話目になる音声の認識結果を捨てることができる．そして，2発話を結合した音声ファイルに対する「音声認識終了」メッセージを受け取ると,「待機状態」に遷移する．その後，音声認識結果を受け取ることで,「システム発話中」状態へと遷移，2発話を結合した音声認識結果に基づくシステム発話を生成する．

2.2 処理の詳細

各発話の録音は，adintool を MMDAgent 本体の Julius と平行に動作させて行っている．この際の adintool の発話区間検出は，Julius と同じパラメータにて行う．

発話を結合する際の「発話の間」は，本稿では 1200 ミリ秒とした．これは，Raux らの研究 [2] を参考に設定したが，今後，本システムにおける適切な値を決定する必要がある．

発話を結合する際に，発話の始終端の一定区間を削除してから結合した．これは，adintool の発話区間検出に

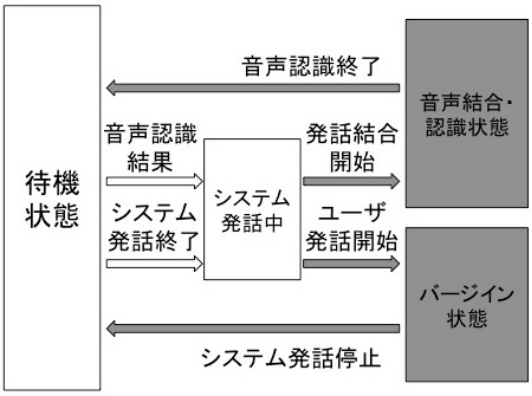

図 3: 状態遷移図

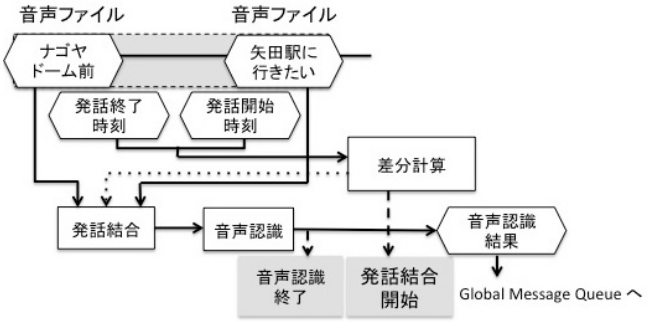

図 4: 処理の流れ

おいて，音声認識時と同様に，発話の前後に一定のマージンをつけて録音が行われるためである．本稿では予備実験の結果，1発話目の音声ファイルの末尾から 290 ミリ秒，2発話目の音声ファイルの先頭から 230 ミリ秒を削除した後，2つの音声ファイルを結合している．なお，2発話目の音声ファイルの先頭部分の削除区間長の方が短い理由は，2発話目冒頭の子音区間を削除してしまわないためであることを，予備実験において実験的に確認している．

結合した音声ファイルの認識には，MMDAgent 本体のものとは別の Julius を使用している．この際の言語モデルや音響モデルなどの設定は，MMDAgent 本体の Julius と同じにしている．

3. おわりに

本稿では，誤分割された2発話を結合して音声認識し直す音声対話システムの実装を行った．今後は，実装したシステムの評価を行う．

参考文献

[1] 李晃伸，大浦圭一郎，徳田恵一: ”魅力ある音声インタラクションシステムを構築するためのオープンソースツールキット MMDAgent”, 電子情報通信学会技術研究報告.SP, 音声 111(365), pp.159–164, 2011.

[2] Antoine Raux, Maxine Eskenazi: “Optimizing Endpointing Thresholds using Dialogue Features in a Spoken Dialogue System”, Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue, pp.1–10, 2008.