Vol.2013-NL-211 No.5
Vol.2013-SLP-96 No.5
2013/5/23

ユーザ発話の誤分割に起因する問題を事後的に修復する音声対話システム

堀田 尚希[1,a)]　駒谷 和範[1]　佐藤 理史[1]

概要：音声対話システムでは，適切な内容の応答とともに，適切なターンテイキングを行えることが重要である．特に素早く応答する音声対話システムを構築する場合，ユーザ発話内の短い無音区間 (言い淀み等) により，(1) システムが誤って応答を開始する，(2) 発話区間が誤って分割され，正しい音声認識結果が得られない，という問題が生じる．本研究ではこの 2 つの問題の事後的な修復を図る．具体的には，まず発話区間が誤分割されているかどうかを判定し，その結果に基づき分断された発話を統合した解釈を行う．次に，対話管理を行う FST に新たな状態を定義することにより，誤って開始したシステム発話を停止する．誤分割された発話の解釈法の性能を評価した結果，単純に 2 つの断片の音声認識結果を接続する手法に比べ，発話の正解率が 12 ポイント上昇した．

1. はじめに

音声対話システムでは，適切な内容を応答するとともに，適切なターンテイキングを行えることも重要である．一般にターンテイキングとは話者交替のことで，二者が交互に話すことを指す．音声対話システムの場合，どのタイミングで，システムが発話を開始し，発話を終了（停止）するかを制御する必要がある．

音声対話システムにおいて適切なターンテイキングを実現するには，システムには以下の挙動が必要である．

(1) ユーザの発話が終了すると速やかに応答を開始する．
(2) システムの発話中にユーザが発話を始めると，すぐに発話を停止する．
(3) ユーザの発話中に話し始めない．

これは，ユーザの発話にシステムが追従する場合を想定している．

不適切なターンテイキングの例を図 1 に示す．図 1 の上の例では，ユーザの発話終了後，システムはすぐに応答しておらず，不適切である．これは，インターフェース設計の観点からも，ユーザの入力に対するフィードバックがなく，適切でない．このため，ユーザの発話終了後，素早く発話を開始させることが必要であり，これを目指した研究やシステムが存在する [2], [4]．一方で，図 1 の下の例では，ユーザの「名古屋にある居酒屋を教えて.」という発話の最中に，システムが割り込んで発話している．これは false cut-in と呼ばれる現象であり，ユーザが発話の途中で言い淀んだ際にしばしば生じる．

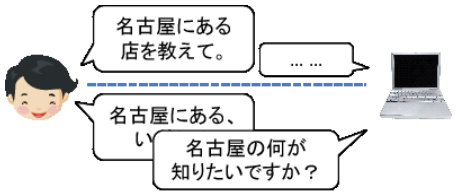

図 1　不適切なターンテイキングの例

本研究では，図 1 の下半分に示されている false cut-in，つまりシステムがユーザ発話の最中に話し始める現象に焦点を当てる．この現象は，言い淀み等の発話中の短い無音区間により，発話区間検出部がユーザ発話を誤って分割することに起因する．このとき具体的には，次の 2 つの問題が生じる．

- 発話断片に対する音声認識誤り
  誤って分割されたユーザ発話の断片に対して音声認識が行われるため，その結果が誤りとなることが多い．
- 不適切なターンテイキング
  ユーザの発話中にシステムが発話を始める．

これは，システムに素早く応答させようとするほど，生じやすい現象である．システムに素早く応答させるには，ユーザ発話の終了判定を素早く行う必要があり，このとき短い無音区間により，誤って発話が終了したと判定さるためである．

本稿では，これら 2 つの問題に対して，事後的な修復を

[1] 名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University
[a)] n_hotta@nuee.nagoya-u.ac.jp

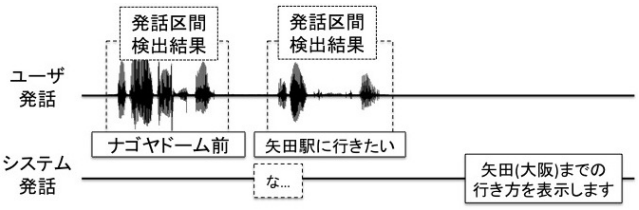

図 2 言い淀みに対する発話の誤分割

図る．具体的には，まず発話区間が誤分割されているかどうかを判定し，その結果に基づき分割された発話を統合して解釈を行う．次に，対話管理を行う有限状態トランスデューサ (FST: Finite State Transducer) に新たな状態を定義することにより，誤って開始したシステム発話を停止する．さらに，誤分割された発話を統合する際に処理待ちが生じた際には，システムがフィラーを生成することにより，不適切な間が生じるのを防ぐ．これらにより，素早い応答を維持しながら，false cut-in を事後的に修復するシステムを実現する．

2. ユーザ発話の誤分割に起因する問題

2.1 発話の誤分割とそれによる音声認識誤り

言い淀み等の，ユーザの発話中の短い無音区間により，発話区間が誤って分割される現象について説明し，それが音声認識誤りの原因となることを示す．言い淀みとは，ユーザが長い単語や文を発話するときに，息継ぎや小休止，あるいは話しながら内容を考えることなどにより発生する，発話内の無音時間のことである．

一般的な発話区間検出 (VAD: Voice Activity Detection) では，音声信号の振幅や零交差数により無音を判定し，その無音の区間長が閾値より長いときに，ユーザの発話が終了したとみなす [1]．この無音区間長の閾値は，発話区間検出でのパラメータのひとつである．例えば音声認識エンジン Julius では，`-tailmargin` というオプションで指定できる．

この無音区間長の閾値を短く設定するのは，素早い応答を実現するための手段の一つである．無音の判定基準（標準的には波形の振幅と零交差数）が同じであれば，この無音区間長を短く設定することで，システムはユーザ発話の終了を素早く検出でき，その結果応答開始を早くできる．

一方でこのとき，発話内の短い無音区間により，誤って発話区間検出が行われることがある．なぜなら，発話内の短い無音区間が，本当にユーザ発話の終了を示すのか，あるいはユーザは言い淀んだだけで，実際には発話を継続しようとしているのか，判定するのが難しくなるからである．この結果，本来は一発話であったユーザの発話が，誤って複数に分割される．本研究では，この現象を発話の誤分割と呼ぶ．

発話の誤分割の例を，図 2 に示す．ここではユーザが，「ナゴヤドーム前矢田駅に行きたい」と発話しようとして，「ナゴヤドーム前」と「矢田駅に行きたい」の間で，言い淀んでいる．「ナゴヤドーム前矢田」は，名古屋市営地下鉄に存在する駅名である．本来ならば，このユーザ発話全体に対して，システムは「ナゴヤドーム前矢田までの行き方を表示します」のような応答をすべきである．

発話区間が誤分割された場合，その音声認識結果は，誤りとなることが多い．なぜなら，ユーザ発話の断片に対して音声認識が行われるからである．図 2 の例では，誤分割された後半の断片「矢田駅に行きたい」に対して音声認識が行われている．この結果，「(大阪府の) 矢田までの行き方を表示します」という応答が行われるが，これは適切ではない．

2.2 発話の誤分割による不適切なターンテイキング

音声対話システムにおいて，発話が誤分割されると，ターンテイキングが不適切になるという過程を説明する．ここでの不適切なターンテイキングとは，ユーザ発話の最中に，システムが発話を開始することを指す．

基本的に音声対話システムでは，音声認識結果が得られると，それに対する応答が開始される．音声認識は，発話区間検出の結果として得られる音声区間に対して行われる．このため，発話区間検出部で発話が誤分割された場合，その誤分割された断片それぞれに対して音声認識が行われる．

このため，発話が誤って分割され，それぞれの区間に対して音声認識結果が得られると，それらに基づきシステムが発話を開始する．この場合，図 2 の例のように，分割された発話の前半部分の終了タイミングは，実際にはユーザ発話の最中である．このため，ユーザ発話の最中にシステムが発話を開始することになる．

発話区間検出結果以外にも様々な情報を用いて，ユーザの発話終了を判定する研究が，これまでにも行われている．ユーザの発話終了の判定は，エンドポインティング (end-pointing) と呼ばれる．佐藤らは，ユーザ発話が終了し，システムがターンを取得すべきかどうかを，決定木学習により判定する手法を示した [5]．その特徴には，音声認識結果の最終語や理解状態，韻律などを用いている．Raux らは，エンドポインティングの性能向上のために，同様に様々な特徴に基づいて，動的に無音区間長の閾値を適応させる手法を示した [2].

これに対して本稿では，発話の誤分割が起こった場合の，事後的な修復に焦点を当てる．つまり，エンドポインティングには，既存の発話区間検出結果をそのまま用いる．ここで既存の発話区間検出とは，音声認識エンジンに付属のものを指す．本稿での提案手法は，特定の音声認識エンジンやエンドポインティング手法には依存しないため，より高性能なエンドポインティング手法と併用することも可能である。

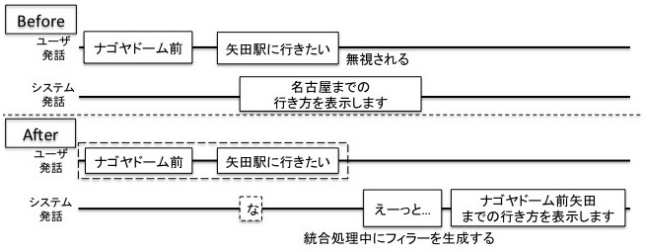

図 3 発話の誤分割が修復される具体例

3. 発話の誤分割を修復するシステム

前章で述べた 2 つの問題を修復するシステムを構築する．これらの問題は発話の誤分割に起因する．

この処理を，本章では以下の 2 つに分けて説明する．

(1) 発話の誤分割に起因する音声認識誤りの修復
(2) ターンテイキングの修復

ここで本研究で構築するシステムの対話例を，本稿で説明する手法の有無を対比しながら示す．まず図 3 の上半分は，ユーザのバージインを受けつけないシステムである．これは，音声認識結果が得られれば応答するという，一般的なシステムの挙動である．ここでは，発話が誤って分割されると，誤分割された断片のうち，前半の断片に対する音声認識結果に基づいて応答が生成される．この例では，「ナゴヤドーム前」という発話断片に対して音声認識が行われ，この音声認識結果に名古屋という駅名が含まれることから，「名古屋までの行き方を表示します」という応答が行われる．この応答は適切ではない．

一方，図 3 の下半分は，本章で説明する手法を実装した場合の例である．ここでは，発話の誤分割が発生した場合，2 発話を統合して解釈を行う．この例では，「ナゴヤドーム前」と「矢田駅に行きたい」という 2 つの発話を統合する．その結果，「ナゴヤドーム前矢田までの行き方を表示します」のような，適切な応答を行うことができる．この際に，前半の断片に対するシステム応答を停止する必要がある．図 3 中の「な」という部分がこれに相当する．なお，発話の統合を行う際には，システムに遅延が生じる．このとき，「えっと…」というフィラーを生成する．これにより，不自然な間が生じるのを防ぐ．

3.1 発話の誤分割による音声認識誤りの修復

2 発話が時間的に近接して得られた場合，それらは元来 1 発話であった可能性がある．元来 1 発話であった場合，それらは誤分割された断片であるため，音声認識結果を修復する必要がある．したがってこの時，それらを統合して解釈を行う．これにより，本来の発話区間に対応する音声認識結果の取得を試みる．

3.1.1 発話の組が本来 1 発話かどうかの判定

発話区間の修復が必要かどうかを判定する．つまり，発話の組が与えられた場合に，それらを統合して解釈すべきかどうかを決める．このために，発話の組が本来 1 発話であったか否かを判定し，本来 1 発話であった発話の組は統合して解釈する．

本稿ではひとまず，2 つの発話の間の間隔を使用して，本来 1 発話であったか否かを判定する．つまり，発話の組に対して，前のユーザ発話の終了時刻と後のユーザ発話の開始時刻の差（以下，発話間間隔）が，ある一定時間以内かどうかで，これを判定する．この値の決定方法は，4.2 節で後述する．

3.1.2 誤分割された発話の統合解釈

発話の組が本来 1 発話であり，誤分割されたものであると判定された場合，この 2 発話を統合して解釈する．ここでは以下の 2 種類を考える．

(1) 2 発話の音声ファイルを結合し，再度音声認識を行う．
(2) 2 発話それぞれに対する音声認識結果の文字列を，単純に繋げて解釈する．

この 2 種類の方法は，結合処理をした後の音声認識結果の信頼度に応じて選択する．

まず，誤分割された 2 つの音声ファイルを結合し，再度音声認識を行う．ここでは，発話区間検出結果に相当する音声ファイルを，逐次保存している．この音声認識により，発話を結合した場合の音声認識結果を得る．図 4 に，誤分割された発話を結合する際の処理の流れを示す．

この際，発話区間検出で付与されたマージンを削除してから結合する．発話区間検出では，その結果を音声認識に使用することから，発話の前後に一定のマージンが付与されるため，このマージンを削除してから結合する．ここでは，前の断片の文末の無音区間 (silE) と，後ろの断片の文頭の無音区間 (silB) に相当する部分を削除する．これらの無音区間は，音素アライメントの結果から得る．

この後，結合した音声ファイルに対する音声認識結果の信頼度を使用する．ここで信頼度として，Julius が出力する単語ごとの信頼度の，文全体での平均値を使用した．もしこの信頼度が高い場合には，音声ファイルを結合して得た音声認識結果を使用する．一方，低い場合は，結合して得た音声認識結果を使用せず，結合前の音声認識結果の文字列を，単純に繋げて用いる．

3.2 ターンテイキングの修復

適切なターンテイキングの実現に必要な条件は，以下の 2 点である．

- ユーザが発話中の場合，システム発話を停止する．発話が誤分割された場合，ユーザ発話が継続中であるにも関わらず，その前半の断片に対する応答を開始してしまうため，これが必要である．
- 音声認識結果の修復の際に遅延が生じた場合，システムが何らかの反応を行う．

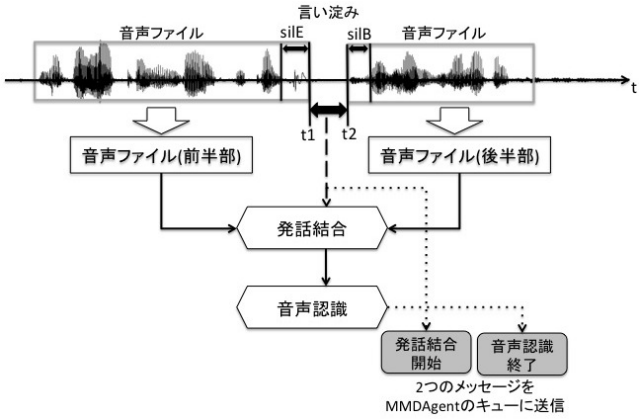

図 4 発話の結合処理

本稿での提案手法は，MMDAgent[4] のプラグインとして実装する．この対話管理部は，有限状態トランスデューサ (FST: Finite State Transducer) を用いて実現されている．このため，この FST を拡張することで，発話の誤分割が生じた際の，ターンテイキングの修復を実現する．さらに，2 発話の統合処理を行う際に，適切なシグナルを FST に送ることにより，フィラーを生成し，不自然な間が生じることを防ぐ．

本節では，まず準備として，MMDAgent について説明する．続いて，ユーザ発話中に，誤って開始したシステム発話を停止する方法と，音声認識結果の修復の際に遅延が生じた場合，何らかの反応を行う方法について，順に説明する．

3.2.1 MMDAgent とその FST

MMDAgent は，名古屋工業大学で開発された，オープンソースの音声インタラクションシステム構築ツールキットである．MMDAgent は，応答のリアルタイム性にも配慮して実装されている．このため本研究では，ユーザ発話に対し素早く応答するという，ターンテイキングに関する基本的な条件を満たすことから，これをベースに実装を進める．

MMDAgent では，全ての機能がモジュール化され，プラグインとして実装されている．本稿での提案手法も，このプラグインの一つとして実装する．プラグイン間の通信は，Global Message Queue と呼ばれるキューで，メッセージをやり取りして行う．このキューに送られたメッセージをトリガーとして，状態遷移が行われる．

対話管理を行う FST の例を図 5 に示す．図 5 中の四角は状態を，矢印は状態遷移を示す．各状態遷移には，その状態遷移が起こる条件と，状態遷移とともに実行する処理内容を，「(遷移条件)/(処理)」という形式で付与している．遷移条件の ϕ は無条件で遷移すること，処理の ϕ は処理無しで遷移することを意味する．

図 5 は，MMDAgent に付属している，サンプルの FST の状態遷移の概要を表す．この FST は，「音声認識結果が得られれば応答する」という，一般的な音声対話システムの対話管理規則を表している．この FST は 2 つの状態から成る．まず「待機状態」は，システムがユーザの発話を待っている場合と，ユーザが発話している場合を表す．「システム発話中状態」は，システムが発話をしている状態を表す．状態遷移の説明は以下である．

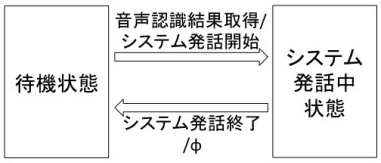

図 5 一般的な音声対話システムにおける状態遷移

- 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の発話を開始し，システム発話中状態に遷移する．
- システムの発話が終了したら，再び待機状態に戻る．

つまり，システムの発話中に，別の音声認識結果が得られても，何も動作しないことになる．

3.2.2 システム発話を停止するための状態の追加

前節で述べた FST は，ユーザが話している最中は，システムが発話を停止するという要件を満たさない．システム発話中に，ユーザの発話が開始されたとしても，それを条件とする状態遷移がないからである．

ユーザが発話を開始すればシステムは発話を停止するという条件を満たすために，この FST にバージインを表す状態を追加する．ここでバージインとは，システム発話中にユーザが割り込んで話し始めることである．誤分割された発話の前半の断片に対してシステムが応答を開始した場合も，ユーザ発話の後半の断片の開始を検知して，システム発話を停止できれば，ターンテイキングを修復できる．

バージイン状態を追加した後の状態遷移を，図 6 の右上部分に示す．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て，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に，システムの発話を停止させる機能が実現できる．

3.2.3 発話の統合処理時におけるターンテイキング

3.1 節で述べた発話の統合処理の間にも適切なターンテイキングを行う必要がある．これには具体的には，次の 2 つの機能が必要である．

- 誤分割された発話の断片に対する音声認識結果が得られるが，これに基づくシステム発話を停止する．
- 発話の統合処理中に遅延が生じた際に，何らかの反応を行う．

このためにプラグインに以下を実装する．

(1) 発話結合と音声認識処理を管理するメッセージの送信
(2) 発話結合と音声認識処理を示す状態の追加

まず，2 つのメッセージを MMDAgent のキューに送信

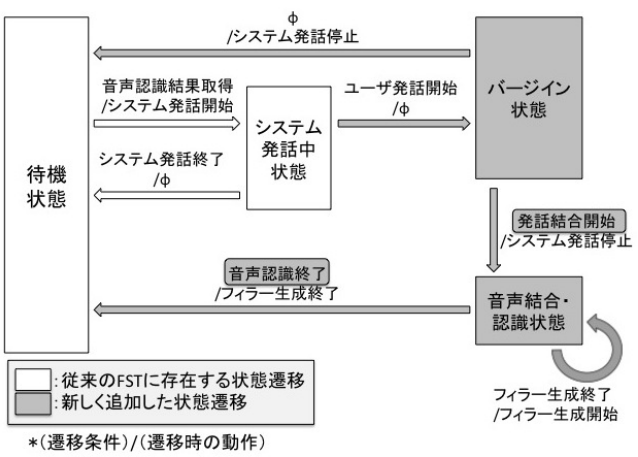

図 6 提案手法を実装した FST

する．2 発話を統合すべきと判定された場合（3.1.1 節）と，結合した音声ファイルに対する音声認識結果が得られた場合を示すメッセージである．この 2 つを，図 4 中に，「発話結合開始」「音声認識終了」というメッセージとして示している．「発話結合開始」メッセージは，誤分割された発話の，後半の断片の開始時刻が得られ次第，発行する．

次に，FST に音声認識・結合状態を追加する．この状態とそれに付随する遷移を，図 6 の右下部分に示す．「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，この音声認識・結合状態に遷移する．これにより，誤分割された発話の，前半の断片に対するシステム発話を停止する．

この音声認識・結合状態には，結合した音声ファイルに対する音声認識結果が得られるまで，つまり「音声認識終了」メッセージを受け取るまで，留まる[*1]．この間，システムは音声認識結果に基づく応答を行えないので，「えっと…」というフィラーを生成する．これにより，発話の結合処理中に，不自然な間が生じるのを防ぐ．

その後，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，待機状態に遷移し，すぐに「音声認識取得」メッセージを受け取ることで，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．なお，誤分割された発話の，後半の断片に対する音声認識結果が，音声結合・認識状態に留まっているタイミングで得られるが，それに対応する状態遷移はない．この結果，後半の断片の音声認識結果について，システムが応答を行うことも防いでいる．

4. 評価実験

本章では，別途収集したデータを用いて行った，提案手法の評価について述べる．ここでは，3.1 章 で述べた発話の誤分割による音声認識誤りの修復部分について，評価を行う．タイミングの観点からの評価は今後の課題である．

まず，使用する発話データについて述べ，その後，発話を結合する条件や統合解釈時のパラメータを決定するために，調査や実験を行う．さらに，この音声データに対して音声認識を行い，発話の正解率を求め，提案する発話解釈法の有効性を示す．

4.1 対象データ

レストラン検索システム [3] を用いて収集された対話音声データを，対象データとして用いる．この音声データは，30 名から 4 対話ずつ，合計 120 対話を収集したものである．本稿では，1 対話ごとに録音されていた音声ファイルに対して，データ収集時と同等のパラメータ設定で音声認識を行い，音声認識結果とその発話タイミングを得た．このときの言語モデルは，語彙サイズ 39900 の統計的言語モデルを使用した．この結果，得られた発話（発話区間検出結果）は合計 6615 発話であった．この中には，短い雑音音声なども含まれる．

本稿では，発話の誤分割の修復が必要である可能性のある発話の組を対象として，評価を行う．修復が不要な発話の組に対しては，提案手法では何も処理を行わないため，音声認識結果には影響しないためである．具体的には，以下の条件を満たすものを，評価対象データとする．

- 2 つの発話区間検出結果が時間的に近接しているもの
- 雑音ではないもの（ユーザ発話が含まれるもの）

まず，2 つの発話区間検出結果が近接しているという条件を，発話間間隔が 2000 ミリ秒以下とした．発話間間隔は，前のユーザ発話の終了時刻と，後のユーザ発話の開始時刻との差を指す．つまり，2000 ミリ秒以上間隔が開いている場合は，元来 1 発話であった可能性はなく，修復は必要ないとみなした．この値は，データを見ながら経験的に設定した．

次に，時間的に近接していても，発話の組の一方が雑音であるものも除外する．雑音を含むものの判定は，単純に，発話長が 800 ミリ秒以下であるとした．これは，データを収集する際にも，Julius のオプション `-rejectshort` で，発話長が 800 ミリ秒以下の入力を棄却していたことに相当する．

以上の 2 つの条件を満たす，近接した発話対として，376 組を次節の分析で用いる．

4.2 元来 1 発話であったとみなす発話間間隔の決定

3.1.1 節で述べた，1 発話とみなすべき発話間間隔の閾値を，前節の結果得られた 376 組の発話対を用いて決定する．このために，この全ての発話対に対して，それらが本来 1 発話であったかどうかを，人手で音声データを聴取しながら付与した．

[*1] 現状のプラグインの実装では，結合する音声ファイルの長さ分（1〜2 秒程度）の遅延が発生する．これは誤分割された後半の音声ファイルが得られた後に，結合し，認識を行っているためである．

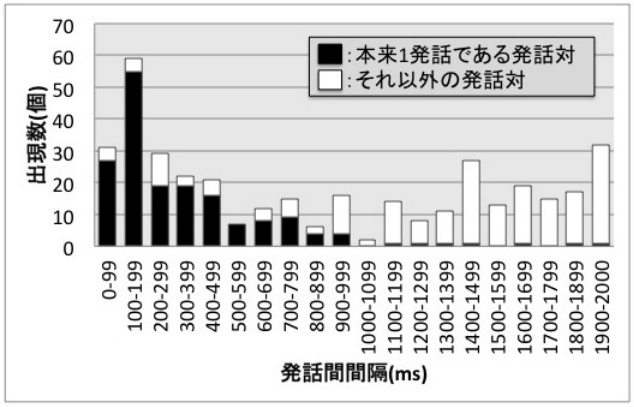

図 7 発話間間隔と 1 発話であるか否かの関係

結果を図 7 に示す．図 7 の縦軸は発話対の出現回数，横軸は発話間間隔を 100 ミリ秒ごとにまとめたものである．図 7 より，発話間間隔が短い発話対は，本来 1 発話であったものが多い．一方，発話間間隔が長い発話対はそれ以外，つまり別の発話であった場合や，片方が雑音であった場合が多い．

この図 7 に基づき，閾値を決定する．これは，閾値未満は本来 1 発話であったと判定した場合に，判定誤りが最少となるよう決定する．具体的には，100 ミリ秒ごとの閾値に対して，本来 1 発話であるものの再現率と適合率を計算し，その調和平均である F 値が最大となる閾値を選ぶ．この結果，閾値を 900 ミリ秒としたときに，最大の F 値 0.87 が得られた．以降，この 900 ミリ秒を閾値として，本来 1 発話であったかどうか，つまり，発話の誤分割が生じていたかどうかを判定する．このような発話対は 202 組存在した．

4.3 統合解釈時のパラメータの決定

4.3.1 音声認識の正解判定

前節までで得られた発話対に対して，音声認識実験を行う．音声認識の正解は，発話ごとに判定する．具体的には，正解データ中のキーワードが音声認識結果に含まれていた場合，その発話を正解とする．正解データにキーワードが複数含まれていた場合，それらが全て含まれていた場合のみを正解とする．キーワードには，地名や店名，駅名など，合計 2789 単語を設定した．正解データは人手による書き起こしから作成した．

前節で述べた 202 組のうち，正解にキーワードを含むものは 153 組であった．以降，この 153 組に対して，音声認識実験を行う．

4.3.2 マージンの削除方法

3.1.2 節で述べたように， 発話を結合して再度音声認識を行う前に，発話断片の前後を削除してから発話を結合する．このマージンの削除方法について検証する．

具体的には，以下の 3 つの削除方法を行った場合の，音声認識の正解数を比較した．

(1) 発話の前後を削除しない．
(2) 発話の前後を一定値だけ削除する．
(3) 発話の前後を無音時間分だけ削除する．

1 番目の方法では，発話対をそのまま結合する．2 番目の方法では，発話の前後を一定値だけ削除する．ここでは，前の断片の末尾を 290 ミリ秒，後の断片の先頭を 230 ミリ秒だけ削除した．これらは予備実験から得た値である．3 番目の方法では，発話の前後を無音時間分だけ削除する方法である．前の断片の文末の無音区間 (silE) と，後ろの断片の文頭の無音区間 (silB) に相当する部分を削除した．これらの無音区間は，音素アライメントの結果から得た．

結果を表 1 に示す．表より，無音区間長を削除する場合が，最も音声認識が正解であった発話数が多いことがわかる．以降の音声認識実験では，発話を結合する際には，無音区間長分を削除してから結合する．

表 1 結合前のマージンの削除法による音声認識の正解率

発話の結合法	正解である発話数
削除しない	98/153 (64%)
一定値だけ削除	105/153 (69%)
無音区間長だけ削除	114/153 (75%)

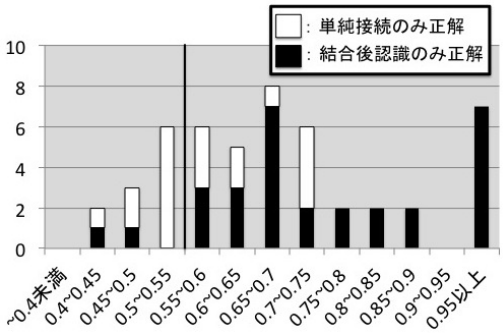

図 8 音声認識信頼度ごとの統合解釈方法の正誤

4.3.3 結合解釈方法を選択する単語信頼度の閾値

発話の統合解釈を行う際に，結合処理をした後の音声認識信頼度に応じて，解釈法を変更する（3.1.2 節）．本節では，これが結果の選択に有用かどうかを検証する．ここでは，以下の 2 つの方法を比較する．

単純接続 発話断片に対する 2 つの音声認識結果の文字列を単純に繋げる．

結合後認識 2 つの音声ファイルを結合し，再度音声認識を行う．

2 つの方法を選択する尺度として，予備実験の結果，音声ファイル結合後の音声認識結果の信頼度を用いる．具体的には，音声認識結果の信頼度として，各単語の信頼度の平均を用いた．

4.3.1 節で述べた 153 組の発話対に対して，音声認識実験を行ったところ，49 組の発話対において，この 2 つの手法による結果が異なっていた．この結果を図 8 に示す．図 8

表 2 発話の解釈法による音声認識の正解率

発話の統合方法		正解発話数
統合しない	(1) 前半断片のみ	43/153 (28%)
	(2) 後半断片のみ	31/153 (20%)
統合	(3) 2 つの認識結果を接続	103/153 (67%)
	(4) 音声ファイルを結合	114/153 (75%)
	(5) 提案手法	**121/153 (79%)**

の横軸は発話結合後の音声認識信頼度，縦軸はその出現数を示す．図より，比較的信頼度が低い部分に，“単純接続”でのみ正解であった発話対が多いことが確認できる．この結果，最適な閾値を 0.55 とし，以降で用いる．つまり，結合後の音声認識信頼度が 0.55 より高い場合には，その音声認識結果を用い，低い場合は，元々の 2 つの音声認識結果を繋げて用いる．

4.4 発話の解釈法の評価

誤分割された発話の統合解釈方法を評価する実験を行った．対象は 4.3.1 節で述べた 153 組の発話対である．

具体的には，以下の 5 つの発話解釈法による，発話単位での音声認識の正解数を比較した．方法 (1) と方法 (2) では，誤分割された発話の統合解釈を行わない．方法 (3) から方法 (5) は，統合解釈を行うが，その方法が異なる．

(1) 前半の発話断片の音声認識結果を用いる．
(2) 後半の発話断片の音声認識結果を用いる．
(3) 2 つの音声認識結果の文字列を単純に繋げる．
(4) 2 つの音声ファイルを結合し，再度認識を行う．
(5) 単語信頼度に応じて結合方法を選択する．

方法 (1) は，ユーザのバージインを受け付けないシステムに相当する．つまり，前半の発話断片に対するシステム応答中に存在する，後半のユーザ発話は無視される．方法 (2) は，ユーザのバージインを受け付けるシステムにほぼ相当する．つまり，前半の発話断片に対する応答は，後半の発話断片の入力により中止される．このため，後半の発話断片に対する応答のみが，ユーザに対して出力される．方法 (3)，方法 (4) は，4.3.3 節での 2 つの方法（単純接続，結合後認識）である．方法 (5) は，4.3.3 節の結果を用いて，方法 (3) と (4) を選択した場合である．

結果を表 2 に示す．まず，発話を統合して解釈するか否かに着目する．例えば，提案手法 (5) と方法 (1) の正解率を比較すると，前者が 51 ポイント高かった．ここでは元来は 1 発話であった可能性が高い発話対を評価対象としているものの，発話を統合して解釈することで，正しい音声認識結果が得られる発話対が多く存在することが示されている．このことより，誤分割された発話を統合解釈する必要性が示されている．

次に，2 発話を統合して解釈する各方法の中での違いに着目する．単純に 2 つの音声認識結果を接続する手法 (3) と比べて，提案手法 (5) では，発話の正解率が 12 ポイント上昇していた．このことから，誤分割された発話に対して，音声ファイルを結合して，再度音声認識を行うアプローチの有用性が示されている．さらに，手法 (3)(4) と提案手法の性能を比較することで，音声認識結果の信頼度に応じて，結合解釈方法を選択するアプローチの有効性も示されている．ただし本稿では，選択に用いる音声認識信頼度の閾値を closed test で決めているため，異なるデータ，もしくは交差検定で，閾値を決定して評価する必要がある[*2]．

得られた音声認識結果の具体例を，表 3 に示す．これらはそれぞれ，方法 (3)，方法 (4) のいずれかで正解であった例を示している．[] 内の単語はキーワードを，<s>は発話が分割された箇所を示している．方法 (1) や (2) による結果は，方法 (3) による結果のうち，<s>の前後の，いずれかの部分のみを取り出した場合に相当する．

表 3 の上半分は，方法 (3) でのみ正解となった例である．1 つ目の例は，ユーザが “やんやしゃくじ” という店名のみを言い直した発話である．統合しない場合は，前半の発話断片の音声認識結果は無音 (認識結果: <silB>,<silE>) であり，後半の発話断片に対しては正しい音声認識結果が得られていた．一方，統合した場合（方法 (4)）では，後半の発話断片に対する音声認識結果が，誤りに変化している．2 つ目の “タバコは吸えるか，禁煙席” という発話では，統合しない場合は，両方の発話断片に対して正しい音声認識結果が得られていたが，統合した場合では，後半の発話断片に対する音声認識結果が，誤りに変化している．これらは，発話を結合することによる副作用である．つまり，無音区間を削除したことや，前の断片と統合解釈する際に言語モデル確率が変化することで，音声認識に誤りが生じる可能性を示している．

表 3 の下半分には，方法 (4) でのみ正解となった例を示している．1 つ目の “まめぞんえすかてん” は，「マ・メゾン」という洋食屋の，エスカという場所にある支店の名前である．これは，この文字列全体で一つのキーワードとして登録されており，途中で言い淀み等により誤分割された場合には，それぞれの断片が単語辞書にないため，正しく認識される可能性はない．このように，キーワードの途中で言い淀んだ場合などには，それを検出し，結合して解釈するというアプローチが有効である．図 2 での「ナゴヤドーム前矢田」駅も同様の例である．2 つ目の “タバコを削除” という発話では，前半の断片に対する音声認識結果を，後半の断片と結合させて再度音声認識を行うことにより，正解へと変化している．これは，後半と統合して解釈することにより，発話が長くなり，言語モデルの制約がうまく効いた例である．

[*2] ただし，このしきい値は，データに強く依存はしないと考えている．

表 3 音声認識結果の例（<s>は発話が分割された箇所，[] 内はキーワード.）

方法 (3) の結果	方法 (4) の結果	正解
<s> [やんやしゃくじ]	やはり <s> [佐屋] 焼い百	<s> [やんやしゃくじ]
[タバコ] は吸えるか <s> [禁煙]	[タバコ] は吸えるか <s> いいん	[タバコ] は吸えるか <s> [禁煙] 席
豆 [うどん] <s> えす勝手	[まめぞん <s> えすかてん]	[まめぞん <s> えすかてん]
割烹凍っ <s> [削除]	[タバコ] を <s> [削除]	[タバコ] を <s> [削除]

5. 結論

本研究では，適切なターンテイキングを実現するために，素早く応答するシステムにおいて問題となる，発話の誤分割に起因する 2 つの問題を挙げた．1 点目の問題は，誤分割された発話断片に対する音声認識誤りである．本研究ではこれに対して，発話の組が本来 1 発話であると判定された場合に，この 2 発話を統合して解釈する手法について述べた．2 点目は，発話の誤分割により，システムが間違って話し始めることである．本研究では対話管理を行う FST を拡張することで，発話の誤分割が生じた際の，ターンテイキングの修復を行った．

1 点目の音声認識誤りの修復について，発話ごとの音声認識の正解数により評価した．その結果，ユーザのバージインを受け付けないシステムと比べ，発話の正解率は大きく向上した．さらに，単純に音声認識結果の文字列を繋げる方法と比べても，発話の正解率が向上することを示した．

今後の課題として以下が挙げられる．

(1) 誤分割された発話断片に対する音声認識性能という観点でシステムの評価を行ったが，ターンテイキングという観点からの評価も必要である．

(2) システムの動作を決めるパラメータなどの評価や検証が必要である．具体的には，本稿では，元来 1 発話であったか否かを，発話間間隔のみで定めた．また，発話の統合解釈法も，結合後の音声認識信頼度のみを用いて行った．これらに関して，他の有効な判定方法を検討するとともに，異なるデータ，もしくは交差検定で，性能を検証する必要がある．

(3) 発話を結合する部分の実装を改良する必要がある．現在の実装では，発話区間検出結果が得られた後に，統合の必要性を判定し，音声ファイルを結合した後，再度音声認識を行っている．現在は，システムがフィラーを生成することでその遅延時間を繋いでいるが，この部分をオンライン実装化すれば，結合処理による遅延を大幅に短縮できる．

参考文献

[1] Benyassine A., Shlomot E., Su H. Y., Massaloux D., Lamblin C., and Petit J. P. ITU-T recommendation G.729 Annex B: a silence compression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications. *IEEE Communications Magazine*, Vol. 35 (9), pp. 64–73, 1997.

[2] Antoine Raux and 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.

[3] 西村良太, 駒谷和範. データベース検索音声対話システムにおける対話状態の推定. 情報処理学会研究報告, Vol. 2012-SLP-90, No. 20, pp. 1–7, 2012.

[4] 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報告 SP, 音声 111(365), pp. 159–164, 2011.

[5] 佐藤玲, 東中竜一郎, 田本真詞, 中野幹生, 相川清明. 決定木学習による音声対話システムのための話者交代判定法. 情報処理学会研究報告, Vol. 2002-SLP-44, pp. 43–48, 2002.