TR-IT-0250

意味的類似性を用いた音声認識結果からの正解部分特定法および特定部分のみ翻訳する部分翻訳手法について

脇田 由実　飯田 仁
Yumi WAKITA　Hitoshi IIDA

1997 年 12 月

概要

自由発話文の音声認識誤り文を解析するために，なるべく文法的制約を用いず、用例と入力文との単語間の意味的類似性を用いることで認識誤り文から正しく認識された部分を特定する方法を提案し、本特定法にて特定された部分のみを翻訳する音声翻訳システムを構築した。ここでは、まず正解部分特定法について説明し、正解部分特定率、特定された部分からの文理解度、部分翻訳結果の翻訳文理解度、を用いて評価した正解部分特定法の評価結果を報告する。最後に、本特定法を用いた部分翻訳システムの詳細と翻訳文評価ツールについての資料を添付する。

エイ・ティ・アール音声翻訳通信研究所
ATR Interpreting Telecommunications Research Laboratories

目次

1 はじめに **1**
- 1.1 本研究所での研究の位置付け . . . 1
- 1.2 研究の背景 . . . 1

2 音声認識結果正解部分特定について **3**
- 2.1 Constituent Boundary を用いた自由発話文の解析 . . . 3
- 2.2 正解部分の特定方法 . . . 3
- 2.3 正解部分の特定手順 . . . 4

3 正解部分特定法を導入した翻訳システム **5**

4 正解部分特定法の評価 **6**
- 4.1 正解部分特定率 . . . 6
- 4.2 特定部分文からの文理解率 . . . 8
- 4.3 翻訳正解率に対する効果 . . . 8
- 4.4 考察 . . . 9
  - 4.4.1 表現パターン間の挿入誤り . . . 10
  - 4.4.2 文末表現の誤り . . . 10
  - 4.4.3 N 個以上の単語からなる不自然な表現への誤り . . . 10
  - 4.4.4 学習データに稀な表現への誤り . . . 11
  - 4.4.5 複数文からなる発話文の翻訳 . . . 11
  - 4.4.6 正解部分のみを翻訳することによる誤訳 . . . 11
- 4.5 形態素体系の違いによる正解部分特定性能の違い . . . 12

5 おわりに **13**

6 著者の関連発表資料 **16**

7 参考資料 (ツール関連) **17**
- 7.1 正解部分特定法を導入したデモ用日英、英日音声翻訳システム . . . 17
  - 7.1.1 97 年度デモシステム概要 . . . 17
  - 7.1.2 インストールと設定 . . . 18
    - インストール . . . 18
    - 設定 . . . 19
  - 7.1.3 操作説明 . . . 20
    - 起動および終了方法 . . . 20
    - 画面操作の詳細 . . . 20
  - 7.1.4 実装説明 . . . 21
    - プログラムファイル . . . 21
    - 設定ファイル . . . 21

サブプロセスの呼び出し 22
音声合成処理について 22
7.2 翻訳結果評価ツールと作業手順 22
7.2.1 評価作業手順 22
7.2.2 評価ツールの起動 ー圧縮から結果を出すまでー 24

第 1 章

はじめに

1.1 本研究所での研究の位置付け

高性能な音声翻訳システムを構築するためには、その基本システムである音声認識部、言語翻訳部、音声合成部などが高性能であることが必要条件である。本研究所では、従来、まずは各基本システムの性能を向上させることに重みを置き、研究開発を進めてきた。たとえば音声認識については、HMMを基本とした音響モデルの構築、話し言葉を意識した統計的言語モデルの構築を行ない、それらの性能を音素認識率や単語認識率で評価してきた。言語翻訳については、文法的に記述が困難な話し言葉の解析と翻訳を可能にするために、用例から多くの話し言葉の言い回しなどを学習し、それを用いた変換主導型の翻訳システムを構築し、翻訳文の正確さや自然性を評価してきた。次のステップとして、音声翻訳システムを実現するためには、さらに各システムが互いの規則や知識を活かしながら、またはお互いの欠点を補いながら処理を行なうしくみを導入し、各システムを統合した完成度の高い音声翻訳システムを構築する必要がある。この観点から、私は、以下のしくみを新たに音声翻訳システムに導入することが必要だと考えている。

音声認識部については、これを言語翻訳部への入力文作成部と考えると、必ずしも認識結果の音素認識率や単語認識率が100% である必要はなく、むしろ次の条件を満たすことが重要である。

(1) 言語翻訳に必要な言葉が認識されていること

(2) 誤訳を招くような単語に誤認識しないこと

また言語翻訳部では、音声認識が度々認識誤りを起こす現実を考慮し、誤りを含んだ文の処理が必要になる。具体的には、

(3) 誤認識単語を含んだ認識結果文の解析が可能であること

(4) 誤認識結果に対応した誤訳をしないこと

が重要である。

上記 4 つの中の (3)(4) を満たす言語翻訳技術を確立するための取り組みとして、自由発話文における音声認識誤り文をなるべく正しく翻訳するための言語翻訳手法の研究を行なってきた。本論ではこの研究成果を報告する。

1.2 研究の背景

連続音声認識において，N-gram と呼ばれる統計的手法に基づいた言語モデルが広く使用されており[1]、限られた探索空間上で認識精度を向上させるためには，信頼できる単語連鎖統計値を得るための大量のデータを用いて、大きなN値に設定された N-gram を用いるのが効果的である。しかし、実際には大量のデータを使用することは非現実的であり、結局、比較的小さいN値である bi-gram や tri-gram を用い，単語の局所的な連鎖にのみ制約を与えて使用しているのが現状である。従って、単語 N-gram モデ

ルを用いた認識では，度々，N単語以上からなる大局的な部分からみれば不自然な文を認識誤り文として出力する．音声対話や音声翻訳システムを実現するためには，上記のような音声認識誤り文を扱うための言語処理が必要であると考える．

従来、文脈自由文法に則って非文法的な文を解析する手法が提案されており[2; 3]、一部の音声認識誤り文の解析にも有効であることが確認されている．また，それを音声翻訳システムに導入した例も紹介されている[4]。これらは，入力文中に解析できない部分があったとき，その部分を削除、あるいは他の単語を挿入，置換しながら解析を続行することにより，音声認識誤り文の解析を可能にしている。しかしこの方法は，基本的には文全体の文法記述が可能であることを前提とし、その中での一部分の誤りにのみ対応できるものである。実際の自然発話文に頻繁に出現すると思われる文全体の文法記述が困難な文には十分に解析できないのが問題となる。

一方、文全体の文法記述が困難であると思われる自然発声文の解析を可能とするために，文全体を部分文に分け、各々の部分文を部分木で記述し、この部分木を列挙したもので文全体を記述する方法も提案されている[5]。　この方法は自由発話文の解析を可能とする上で効果的な方法である。しかし，上記部分木もN-gramモデルと同様に、局所的な部分文の範囲で解析を行なうものであり，認識処理で不足している「大局的にみた言語的制約」を補うものではない。従って，局所的には既に制御されている認識誤り文を誤りであると判断できず誤ったまま解析してしまうという問題があると思われる。

さらに，これら従来の解析方法は文脈自由文法による文法的制約を基本としているが、意味的な整合性を判断した解析ではないため、文の「不自然さ」を判断するには不十分であると考える。

我々は、自由発話文の音声認識誤り文を解析するために，なるべく文法的制約を用いずに、認識誤り文から正しく認識された部分を特定し、特定されなかった部分を必要に応じて修復しながら解析を行なうことが必要であると考えている。本論では、その第1歩として、文の大局的な部分を対象に、その意味的な自然性を判断することにより，認識結果文から正しく認識された部分のみを特定する方法を提案し、それを用いた翻訳システムについて述べる。

以下、2章でこの正解部分特定法について述べ，3章で本正解部分特定法を用いた日英及び英日翻訳システムについて述べ、4章で正解部分特定結果および翻訳結果を用いた正解部分特定法の評価結果について報告し、5章でまとめと今後の課題について述べる。また最後に、９７年度研究成果発表会のためのデモシステムとして構築された正解部分特定法による音声翻訳システムと、本システムの評価ツールに関しての資料を参考資料として添付する。

第 2 章

音声認識結果正解部分特定について

2.1 Constituent Boundary を用いた自由発話文の解析

本研究所第3研究室では、自由発話文の翻訳をめざして，TDMT(Transfer Driven Machine Translation) と呼ばれる変換主導型の音声翻訳手法が提案されている[6]。TDMT では、自由発話文に頻繁に出てくる文法記述の困難な言い回しを取り扱うために、実際の自由発話文から表現パターンとその対訳パターンとを学習しておき、実際の翻訳の際には、入力された文に類似した学習表現パターンを選択し、その対訳パターンを用いて翻訳を行なっている。

入力文に類似した表現パターン候補は複数選択されることがある。その中から最も適切な学習パターンを選択するために、学習された表現パターンの変数に相当する単語と入力文の単語間との意味的な類似性を調べ、入力文に含まれる単語と最も類似した単語を含んでいる表現を選択している[7]。この意味的距離は意味シソーラス辞書に従って算出される。この最適表現パターンの探索は、left-to-right のボトムアップ探索処理にて行なわれる。従って、もし文全体に相当する表現パターンが探索できなくても、部分文に相当する表現パターンは選択できる枠組になっている[8]。

2.2 正解部分の特定方法

我々は，上記の CB による解析法で用いた表現パターンと解析結果を用いて，認識結果文から正しく認識された部分のみを特定することを検討している。正解部分の特定に上記解析法を用いる長所を次のように考える.

- 文脈自由文法に基づいた文法規則では，話し言葉も音声認識誤りも同様に解析が困難であるため、両者の区別がつかない場合が多かったが，上記解析法では実際の話し言葉の表現を学習することにより、解析できない部分を音声認識誤りと判断できる可能性がある。
- 上記解析法で扱う表現パターンは，音声認識時に扱えない N-gram のN個以上の単語からなるものもあり，この表現パターン単位で文の自然性を判断することで、N-gram よりさらに大局的な部分を評価できる枠組である。
- 解析をボトムアップに行っているため，文全体が解析できなくても，部分的な解析が可能である。

音声認識の際に考慮されていない「大局的な部分」からの「意味的な不自然さ」を判断するために、具体的には、次のパラメータに注目して正解部分を特定することを考える。

- 入力文と学習文との単語は意味的類似性
- １つの表現パターンの形態素長

この２つのパラメータを用いた正解部分特定条件は次の２つである。

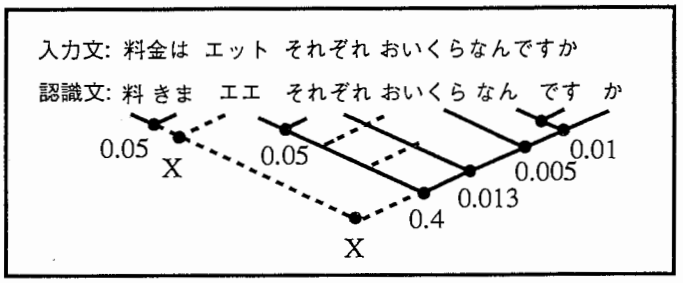


図 2.1: 正解部分特定例

(1) ある入力文中の表現パターンが音声認識誤りを含んでいる場合には，学習パターン内の単語との意味距離値は大きくなると予想される．意味距離値が一定値より小さい表現パターンを正解部分とする．

(2) 単語 N-gram により制御された認識結果は、N 個以下の単語からなる部分については既に制御されており、結果が正解であるか誤りであるかに無関係に「自然」な系列である場合が多い。従って、N+1 個以上からなる範囲から判断して「自然」である場合のみ、正解部分とする。

2.3 正解部分の特定手順

なるべく大局的な部分から優先して正解部分の特定を行なうために、ボトムアップに行なわれた解析結果を、トップダウンに判断しながら特定していく。具体的には、次の手順で正解部分の特定を行なう。

(1) 長い語句の範囲の表現パターンから順にその意味的距離を判断し、意味的距離が閾値より小さい場合は、その範囲に含まれる全ての部分を正解部分とする。

(2) もし意味的距離が閾値より大きい場合には、その部分のどこかに誤りが含まれているとみなして、より小さい部分の表現パターンについて (1) の処理を繰り返す。

(3) この処理が繰り返され、非常に短い語句の範囲での局所的な語句からなる表現パターンを扱うに至る場合は、対象となる短い部分は、他の部分と依存関係がなく文全体から見て不自然な部分であると考える。そこで、解析範囲に含まれている形態素数に下限閾値を設け、解析範囲が細分化されてその形態素数が閾値に達した場合には、意味的距離がたとえ小さくでも、その部分を誤り部分とみなす。

図 2.1に正解部分特定例を示す。これは単語 bi-gram を用いた音声認識結果文を解析した例である。まず音声認識結果に対しボトムアップな解析を行なうことで、いくつかの部分文に対し表現パターンが適応され、結果として、図 2.1に示したような依存関係と意味的距離値が得られたとする。ここでたとえば、形態素数の下限閾値を bi-gram の N より 1 つ大きい 3 、意味的距離の上限閾値を 0.2 とすると、この結果は次のように処理される。この結果では、「料金」が「料決ま」と誤認識しているため、文全体としての解析は失敗している。そこで、解析できた部分で最も大きい表現パターンである「エエ それぞれおいくらなんですか」の意味的距離 0.4 を閾値と比較する。この場合は閾値を上回っているので、この範囲のどこかに誤りがあるとみなし、一段階小さな表現パターン「それぞれおいくらなんですか」を処理する。この部分の意味的距離 0.005 は閾値より小さいので、この部分は全て正解部分と特定する。まだ、判断していない残りの「料決ま」の部分の意味的距離は閾値より小さいものの、この範囲に含まれる形態素数 2 が、下限閾値の 3 未満であるので、この部分は誤った部分とみなされる。

第 3 章

正解部分特定法を導入した翻訳システム

実際の音声翻訳システムＴＤＭＴに上記の正解部分特定法を導入し、特定された部分のみの部分翻訳が可能な音声翻訳システムを構築した。図7.2にシステムの概要を示す。音声認識結果文はまず原言語解析部に入力され、文全体、または部分文の依存関係と意味距離値が出力される。これらを用いて正解部分を特定し、特定された部分文のみを目的言語変換部に入力し翻訳（以後「部分翻訳」と呼ぶ）を行なう。このシステムの詳細は７章に記載する。

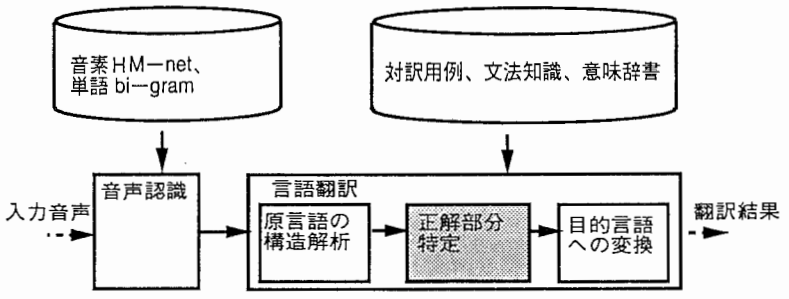


図 3.1: 音声翻訳システム概要

第 4 章

正解部分特定法の評価

このシステムを用いて、本正解部分特定法にて特定された部分文やその翻訳結果文を出力し、これらを以下の観点から評価した。

- 正解部分特定率。
- 特定された部分文のみを提示した場合の全体文に対する文理解率
- 特定された部分のみの翻訳を行なった場合の翻訳文理解率

実験条件は表 4に示す。音声認識手法として、音素HMMと単語 bi-gram 言語モデルを使い、マルチパス探索でワードグラフを出力する連続音声認識方式[9]を用いた。音声認識用の言語モデルにおける形態素体系としてはSLDB 体系を用いており、認識部分から出力された認識結果形態素列を、翻訳部分で用いている形態素体系 (以後 TDMT 体系) に変換するフィルタを通した後、翻訳を行なう。データベースには旅行会話データベース[10]の中の 9 会話分（ 9 話者 119 文）を用いた。本方法は正解部分をトップダウンに特定するため、正しい文が入力され文全体が従来の解析にて成功した場合には、従来と同様、文全体を出力する。特定された部分のみを出力することの優位性を評価するため、評価では、この 119 文を用いて音声認識実験を行ない、その結果誤認識した 70 文のみを評価に使用した。

4.1 正解部分特定率

本方法の正解部分の特定性能を確認するために，正解であると特定した部分に含まれる正解単語の再現率及び適合率を調べ，正解部分の特定を行う前の音声認識結果文の再現率及び適合率と比較した.

$$\text{再現率} = \frac{\text{結果文に含まれる正解単語数}}{\text{正解文の総単語数}}$$

$$\text{適合率} = \frac{\text{結果文に含まれる正解単語数}}{\text{結果文の総単語数}}$$

正解部分の特定に使用した意味的距離の閾値と 1 つの表現パターンに含まれる形態素数の下限閾値とを様々に変えた場合の性能の違いも合わせて評価した．図 4.1、図 4.2に各条件に対する適合率、再現率を示す。次のことがわかる。

音声認識方式	：マルチパス探索ワードグラフ出力型 [清水,1996]
言語翻訳方式	：変換主導型翻訳手法 [古瀬,1994]
翻訳システム	：TDMT1996 年 7 月 ver. + a-filter
音響モデル	：不特定話者用 HM-net、状態数 401、混合分布数 10
言語モデル	：単語 bi-gram、形態素体系 SLDB 体系
データベース	：(認識用) 旅行会話データベース 9 会話 (9 話者 119 文)
	：(評価用) 上記認識用データの誤認識文 70 文

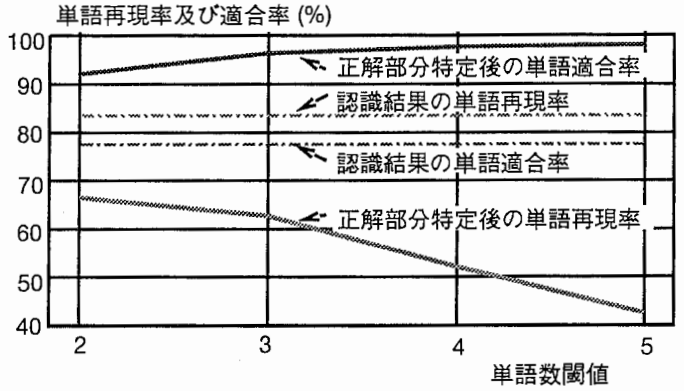


図 4.1: 形態素数の閾値と正解部分特定率との関係

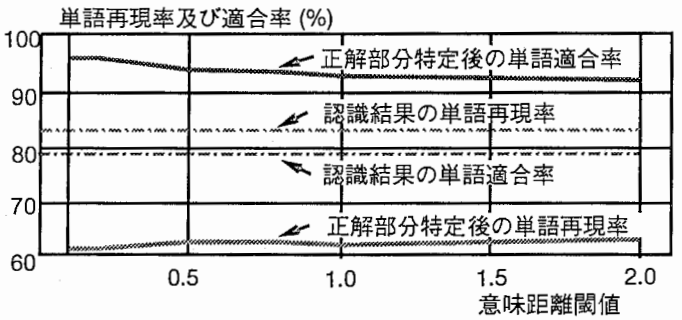


図 4.2: 意味的距離の閾値と正解部分特定率との関係

結果 1-1： どのような閾値の条件下でも，正解単語適合率は 92% を上回っており，特定する前の認識結果と比較しても約 15% 前後向上している．特定した部分が正解である信頼性は高い。

結果 1-2： 正解単語再現率は 20% 前後低下しており，本方法が特定しきれない正解部分も多い。

結果 1-3： 形態素数の閾値については、閾値が大きくなるほど適合率は向上する。特に閾値が 3 以上の場合は、閾値が 2 の場合と比べて適合率は大きく向上している。
本実験では音声認識時に単語 bi-gram を用いており、たとえ誤った単語であっても 2 単語連鎖間だけを取り上げると自然なものが多い。閾値が 2 の場合は誤った 2 単語連鎖を正解部分と特定してしまったことが、3 以上の場合に比べて適合率が低かった原因である。

結果 1-4： 形態素数の閾値が 4 以上になると再現率が極端に低下する。

結果 1-5： 意味的距離の閾値については，その値が小さくなるほど再現率は高く適合率が低くなる傾向がある．しかし，閾値の違いによる性能の違いは、形態素数の閾値を変化させた時に比べて少ない．

結果 1-6： 意味的距離の閾値が 0.2 以下になると適合率は徐々に低下する。

本方法は、学習された表現パターンに基づく解析結果から正解部分を特定ため、たとえ正しく認識された単語であっても、表現パターンと適合しなければ正解部分と特定されず、再現率は低くなっている。しかし、本方法による適合率は高く、これは特定された部分が実際に正解単語である信頼性は高いことを示している。従って、単語 N-gram により制約された認識結果から正解部分を特定するために、学習パターン及び入力パターンに出現する単語間の意味的類似性を用いる本方法は、高い信頼度で正解部分を特定するのに有効であると考えられる。

また、正解部分特定性能は形態素数閾値の変動に対しては敏感であり、たとえば、音声認識時の言語モデルの制約範囲より小さい値に閾値を設定すると適合率は低下する。言語モデルの制約範囲を越えた形態素数閾値を用いることが必要である。

表 4.1: 正解部分特定前後の文理解率の違い

レベル	(L1)	(L2)	(L3)	(L4)	(L5)
特定前	19.6%	22.0%	23.0%	35.5%	0.0%
特定後	20.3%	22.6%	36.8%	15.2%	5.4%

4.2 特定部分文からの文理解率

上記のように正解部分を特定しその部分のみを扱うことが、文全体の意味を理解する上で有効かどうかを確認するために、正解部分特定前及び特定後の各々の音声認識結果文における文理解度を評価した。形態素数および意味距離値の閾値は、前章の実験での最適値（3 と 0.2）を設定した。特定前後の各々の結果文と相当する正解文を比較して、次の 5 段階の評価レベルを結果文に与えることで評価した。評価は日本人 5 名で行なった。評価レベルを以下に示す。

(L1) 正解と比べて同じ意味であると理解できる。

(L2) 少し不自然だが、意味は理解できる。

(L3) 全体の意味はわからないが、部分的には理解できる。

(L4) 間違えた意味にとってしまう。

(L5) 正解部分がない。

各レベルにおける回答数の 5 名の平均値を評価結果を表 4.1に示す。次のことがわかった。

結果 2-1： 本方法にて特定された部分のみ出力することで、誤った意味に解釈される文が半分以上減少した。((L4) が 35.5% → 15.2%)

結果 2-2： 本方法にて特定された部分のみ出力することで、正しく意図を伝える場合は増加したが、その割合は僅かである。((L1) と (L2) の和が 41.6% → 42.9%)

本正解部分特定法は、理解可能な部分を特定し、その部分のみを出力することで、誤った意味への理解を軽減する効果があることがわかった。しかし、正しい意味を伝える効果は僅かである。これは、正しい部分のみ呈示しなくても、評価者が誤った文を自ら訂正しながら読むことで、正しい意味を理解できることを示していると思われる。

4.3 翻訳正解率に対する効果

次に、特定された部分のみを翻訳すろことが、翻訳結果に及ぼす影響を調べるために、正解部分特定前及び特定後の各々の音声認識結果文の翻訳結果文を評価した。閾値の条件は前章と同様に設定した。特定前後の各々の翻訳結果文と正解文の翻訳結果を比較して、次の 5 段階の評価レベルを結果文に与えることで評価した。評価は英会話能力の高い日本人 3 名で行なった。評価レベルを以下に示す。(L1) から (L4) までは、前章の文理解率を評価する評価基準と同じである。(L5) は、特定された部分がない、または翻訳過程で処理が失敗する理由で、翻訳結果が出なかったときに相当する。

(L1) 正解と比べて同じ意味であると理解できる。

(L2) 少し不自然だが、意味は理解できる。

(L3) 全体の意味はわからないが、部分的には理解できる。

(L4) 間違えた意味にとってしまう。

(L5) 翻訳ができない。

各評価レベルにおける回答数の 3 名の平均値を評価結果を表 4.2に示す。次の結果が得られた。

結果 3-1： 従来 11.9% であった翻訳正解率が、本特定法の導入後は約 2 倍の 25.7% に向上している。

表 4.2: 正解部分特定前後の翻訳率の違い

レベル	(L1)	(L2)	(L3)	(L4)	(L5)
特定前	11.9%	0%	0%	2.4%	85.7%
特定後	25.7%	16.7%	26.6%	21.0%	10.0%

結果 3-2： 本正解部分特定法の導入前は、85.2% の文が翻訳できなかったが、本特定法の導入後は、69% ((L1)25.7% + (L2)16.7% + (L3)26.6%) の文に対し、正しいかまたは部分的にも理解できる翻訳文を出力することができるようになった。

結果 3-3： 本正解部分特定法の導入前は、誤った意味に翻訳されることはほとんどなかったが、本特定法の導入後は、誤訳文が 21% に増加した。

以上の結果より、本正解部分特定法は翻訳結果に及ぼす効果は大きく、従来ほとんど翻訳できなかった誤認識文の約 7 割に対し、部分的にも意味を理解できる翻訳文を出力することができた。また、文全体を翻訳できなくても、内容の理解に必要な語が認識されていれば、その部分のみを翻訳することで、ほぼ正しく理解できる翻訳結果が出力可能であることがわかった。この結果は、特定されなかった部分の修復を検討する際には、特定されない部分を全て修復する必要ななく、必要な単語のみの修復で十分であり、場合によっては、修復の必要がないものも多いことをを示唆していると考える。ただし、誤って翻訳してしまった文は問題であり、今後の解決していく必要がある。

最後に次章にて、上記の翻訳結果を分析しながら、本正解部分特定法の音声翻訳に及ぼす効果と悪影響について考察する。

4.4 考察

従来の翻訳でも、誤認識文の 11.9% に対しては正しく翻訳することができている。これは使用した翻訳システム TDMT が、話し言葉に対応するために助詞や一部の文末表現が欠落した言い回しを学習しており、認識誤りがこれらの脱落誤りであった場合にでも正しく解析できたためである。しかし、実際の話し言葉には存在しないような言い回しに誤認識した場合には、翻訳することができなかった。

本正解部分特定法の導入により、効果が見られた誤認識文の特徴は、主に以下のものである。

(a) 表現パターン間の挿入誤り

(b) 文末の言い回しの誤り

(c) N 個以上の単語からなる不自然な表現への誤り

(d) 学習データに稀な表現への誤り

また、誤認識ではないが、従来翻訳できなかった文が、本方法にて翻訳可能になったものに、次ののもがある。

(e) 複数文からなる発話

また逆に、本正解部分特定法を導入することで誤訳してしまった文の主な特徴は、次のものである。

(f) 文末の、人称、立場、肯定文か否定文か、などを決定する表現の誤り

以下に、上記の各々の項目を実際の例を用いて説明する。

表 4.3: 挿入誤りに対する部分翻訳結果例

入力文	電話番号は五二七九です (The telephone number is five two seven nine)
認識結果	電話番号は っお 五二七 あっ 九です
正解部分特定結果	電話番号は .. .. 五二七 .. .. ..
部分翻訳結果	The telephone number .. .. five two seven .. .. ..

表 4.4: 文末での置換誤りに対する部分翻訳結果例

入力文	部屋の予約をお願いしたいんですけれども. (I would like to reserve the room.)
認識結果	部屋の予約をお願いしたい ねすとも
正解部分特定結果	部屋の予約をお願いしたい .. .. ..
部分翻訳結果	.. .. .. would like to reserve the room

4.4.1 表現パターン間の挿入誤り

自由発話文では、「ええ」や「あ」や「ん」などの冗長語が多く話されるため、音声認識では、これらの単語を認識単語として登録している場合が多い。ところが、日本語の冗長語は、短音素で構成されているものが多く、認識時には、他の単語の一部が、度々、冗長語と誤って認識されてしまう傾向がある。さらに、冗長語は文全体のどこで話されるかが予測しにくく、単語 N-gram の枠組では制御しにくい性質がある。これらの理由で、音声認識結果では、この冗長語の挿入誤りが頻繁に起こる。

表 4.3は冗長語の挿入誤りのために、従来の翻訳処理では翻訳できなかった認識誤り文の例である。正解部分を特定して翻訳することで、「電話番号は」や「五二七」などの翻訳結果を出力することができている。ただし、「九です」は、冗長語「あっ」の挿入のために、他の部分とは独立している 2 形態素の局所的な部分であるとみなされ、正解部分と特定することができなかった。

4.4.2 文末表現の誤り

間接的な表現や丁寧な表現が文末で話されることが頻繁にあり、文末の発声が不明瞭な場合は、この部分で認識誤りが起こることも多い。しかし、これらの表現が聞き取れなくても問題なく文の内容が理解できることも多く、そのような場合には、文末に至るまでの部分のみを翻訳することで、ほぼ意味が理解できる翻訳文を出力することができる。

表 4.4は文末での丁寧な表現の部分で音声認識誤りを起こした例である。前半部のみを特定し翻訳することで、意味の理解できる翻訳結果を出力することができている。ただしこの例の場合は、文末表現が人称を決定しているため、主語の「I」を翻訳することはできなかった。

4.4.3 N 個以上の単語からなる不自然な表現への誤り

単語 N-gram における N 個の単語を越えた大局的な範囲で判断した時に、不自然であると思われる認識誤りは、認識時には発見されない。本方法で正解部分を特定することにより、このような誤りを発見

表 4.5: 大局的にみて不自然な表現に対する部分翻訳結果例

入力文	お部屋のご希望はございますか？ (Do you have any preference for the room)
認識結果	親子 のご希望はございますか？
正解部分特定結果	.. gokibou wa gozai masu ka
部分翻訳結果	Do you have any preference ..?

表 4.6: 学習データに稀な表現に対する部分翻訳結果例

入力文	鈴木直子と言います (I am Naoko Suzuki)
認識結果	鈴木直子と い ます (I stay with Naoko Suzuki)
正解部分特定結果	鈴木 直子 と .. ..
部分翻訳結果	.. .. Naoko Suzuki

表 4.7: 文末表現の認識誤りに対する部分翻訳の誤訳例

入力文	都合で泊まれなくなった (I can't stay for some reason)
認識結果	都合で泊まれ ____ なった
正解部分特定結果	都合で泊まれ ..
部分翻訳結果	.. can stay for some reason

できる利点がある。

表 4.5は、入力文の「お部屋」を「親子」と誤って認識した例である。しかしこの例では、単語 bi-gram で制御されている単語の 2 連鎖間の関係には不自然さはなく、文頭に「親子」と話すことも、「親子」＋「の」の連鎖で話すことも不自然ではない。しかし、「親子」＋「の」＋「ご希望」の 3 連鎖でみると、その内容は少し不自然である。本正解部分特定法を用いると、「親子のご希望」の意味的距離値が閾値を越えることから、「親子の」が除去され「ご希望はございますか」のみ翻訳することができる。

4.4.4 学習データに稀な表現への誤り

本翻訳システムは用例主導型のアプローチを採用しており、一般に構造的または意味的に曖昧性を含む文が入力されても、既に学習された表現パターン採用することで、その曖昧性を解消することができる。そのために、認識誤りの結果、学習データには存在しないが一般には不自然ではない文に誤ってしまった場合でも、誤ったまま翻訳は行なわず、正解部分を特定して部分的にのみ翻訳することができる。

表 4.6は、入力文中の「言い」が「い」と誤認識した例である。しかし、認識結果の「鈴木直子といます」は、一般的には自然な文であり、誤りを判断することは難しい。しかし、本旅行案内タスクの学習データには稀な文であるため、意味距離値が閾値を越えてしまい、「鈴木直子と」の部分のみを翻訳することができた。

4.4.5 複数文からなる発話文の翻訳

音声認識結果に誤りがなくても、一発話に複数の文が含まれている場合には、従来の翻訳システムでは文の境界を解析できないため、翻訳結果を出力できなかった。本方法を用いることで、各文を独立に翻訳することが可能となるため、結果的には、文境界を検出して各文を翻訳した場合と同じ結果を出力することが可能となる。たとえば一発話として「わかりました、ありがとうございました」と発声した場合、従来の翻訳システムでは、結果を出力できなかったが、本部分翻訳を行なうことで、「I see, thank you very much」と翻訳することができた。

4.4.6 正解部分のみを翻訳することによる誤訳

文末での、肯定文か否定文か、あるいは、平叙文か疑問文か、を決定する表現が認識誤りを起こした場合、本方法で特定された部分文のみ翻訳することで、違った意味に解釈される翻訳文を出力してしまうことがある。

表 4.7の例は、入力文中の「なく」という否定を表す助動詞が、認識されず欠落してしまった例である。結果的に、「都合で泊まれ」のみが特定され、正しくは、「can not」と否定文として翻訳されるべ

音声認識方式	：マルチパス探索ワードグラフ出力型
言語翻訳方式	：変換主導型翻訳手法
翻訳システム	：TDMT1996 年 2 月 ver.
音響モデル	：不特定話者用 HM-net、状態数 401、混合分布数 10
言語モデル	：単語 bi-gram、形態素体系 TDMT 体系 (間投詞付き)
データベース	：(認識用) 旅行会話データベース 9 会話 (9 話者 119 文)
	：(評価用) 上記認識用データの誤認識文 63 文

表 4.8: TDMT 形態素体系による正解部分特定前後の翻訳率の違い

レベル	(L1)	(L2)	(L3)	(L4)	(L5)
特定前	28.6%	3.2%	4.8%	9.5%	53.9%
特定後	28.6%	18.0%	21.7%	15.9%	16.4%

きものが、「can」と肯定文となり、逆の意味に翻訳されてしまった。

4.5 形態素体系の違いによる正解部分特定性能の違い

上記の一連の評価は、音声認識部での形態素体系として標準化されていた SLDB 体系を採用した場合のものである。SLDB 体系で出力された認識結果は、正解部分を特定する前に、形態素体系変換用のフィルタを通ることで、言語翻訳部の形態素体系である TDMT 体系に変換され、変換された結果から正解部分が特定されている。ＳＬＤＢ体系とＴＤＭＴ体系では、定義されている形態素の品詞分類が違うばかりでなく、1 形態素の単位も異なっているため、上記形態素体系変換フィルタでは、SLDB 体系で定義された単語を分離したり結合したりして、TDMT 体系に形態素変換を行なっている。そのため、誤認識結果については必ず隣接する単語と結合できない単語も多く、その場合は、その単語について形態素変換を行なわず、そのまま単語を出力している。

形態素体系変換フィルタを使用しない場合の正解部分特定の性能を調べるために、音声認識部での形態素体系として TDMT 体系を採用し、形態素体系変換フィルタを通さずに正解部分を特定した場合の評価実験を行なった。

用いたデータベース及び実験条件は 4.5、評価は 4.8に示す。音声認識部、言語翻訳部が、上記の SLDB 体系を用いた実験当時のものから改良されていることと、形態素変換フィルタを通る際に変換できなかった単語を含んでいないことの理由から、正解部分特定法を用いない場合でも、約 32% （レベル (1)+(2)) の誤認識文がほぼ正しく翻訳されている。しかし、60% 以上の誤認識文に対し、翻訳できないか、または誤った翻訳結果を出力するという問題点を、以前として抱えている。本正解部分特定法を用いた場合の各レベルの割合は、SLDB 体系を用いた場合と TDMT を用いた場合はほとんど変わらず、約 47% の誤認識文がほぼ正しく翻訳されており、部分的にも正しく翻訳できた場合を加えると、約 69% の誤認識文で部分翻訳の効果が確認できた。形態素体系によらず、本正解部分特定法が翻訳結果に効果的であることがわかった。

第 5 章

おわりに

自由発話文の音声認識誤り文を解析するために，なるべく文法的制約を用いずに認識誤り文から正しく認識された部分を特定しする方法を提案し、特定された部分のみを部分的に翻訳する音声翻訳システムを構築した。本方法では、正解部分を特定するために、文の大局的な対象として用例と入力との単語間の意味的類似性を算出し、これを用いて特定しているのが特徴である。正解部分特定率、特定された部分から理解できる全体文理解度、特定された部分のみを翻訳した場合の翻訳文理解度などを用いて、本正解部分特定法を評価した。その結果、

- 本方法による特定された部分が実際正解である信頼度は高いこと。
- 特定された部分のみを提示することで、誤り文をそのまま提示した場合に起こる誤理解を大幅に軽減できること。
- 特定された部分のみを翻訳することで、従来翻訳できなかった誤り文の多くが、正しいかまたは部分的にも理解可能な文に翻訳できること。

がわかった。今後は、正解部分特定法を音声翻訳に用いた場合に起きる誤訳を軽減するために、特定されなかった部分の修復の必要性の判断、および修復する方法を検討する必要があると考える。また、本正解部分特定法は基本的には言語に依存しないアプローチをとっており、私がＡＴＲ在籍中には資料化できるほどのまとまった評価ができなかった。しかし、英日音声翻訳においても、本方法が日英翻訳の場合と同様の効果がある感触を得ている。今後多言語間の翻訳における効果も確認する必要があると思われる。

謝辞

最後になりましたが、本研究にあたり、手法の提案当初から御協力頂き、またシステム開発を担当して頂いた現在（株）東洋情報システムの河井淳氏、翻訳結果の評価を担当して頂き、また評価仕様を資料化して頂いた中祢美智子氏、現在マンパワー（株）の栗田めぐみ氏、形態素体系変換フィルタ、原言語解析部、意味的距離算出部を含むＴＤＭＴ翻訳システムを提供頂き、また本研究開発にも有益な御指導頂いた、隅田英一郎主任研究員、現在 NTT （株）の古瀬蔵氏、英語認識システムを提供して下さった現在ＣＭＵの Detlef Koll 研究員、英日翻訳システムへの英語認識部の導入に御協力頂いた第一研究室の Harald Singer 滞在研究員、日本語音声認識システム及び言語モデルを提供して頂いた第一研究室の山本博史研究員、政瀧博和研究員、翻訳結果の評価に御協力頂き、また熱心に討論頂いた音声翻訳研究所の皆様に心から感謝致します。どうもありがとうございました。

1997 年 12 月 25 日

参考文献

[1] L.R.Bahl, F.Jelinek and R.L.Mercer: "A Maximum Likelihood Approach to Continuous Speech Recognition," In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pp.179-190, 1983.

[2] H.Saitou,M.Tomita:"Parsing noisy sentences,", In *proc. of COLING'88*, pp.561–566. 1988.

[3] C.S.Mellish:"Some chart-based techniques for parsing ill-formed input.", In *proc. of the Annual Meeting of the ACL*, pp.102–109, 1989.

[4] A.Lavie, D.Gates, M.GAvalda, L.Mayfield, A.Waibel, and L.Levin:"Multilingual Translation of Spontaneously Spoken Language in a Limited Domain" In *Proc.of 16th ICCL*, pp.442–447, 1996.

[5] T.Takezawa,T.Morimoto:"Dialogue Speech Recognition Method using Rules based on Subtrees and Preterminal Bigrams" In *IEICE Trans in Japanese*, D-II Vol.J79-D-II No.12 pp.2078-2085. 1996.

[6] 古瀬蔵、隅田英一郎、飯田仁:"経験的知識を活用する変換主導型機械翻訳", 情報処理学会論文誌,Vol.35,No.3, pp.414–425, Mar. 1994.

[7] 隅田英一郎、古瀬蔵、飯田仁:"英語前置詞句係り先の用例主導あいまい性解消", 信学論,(D-II),J77-D-II,No.3, pp.5576–565, Mar. 1994.

[8] O.Furuse, H.Iida:"Incremental Translation Utilizing Constituent Boundary Patterns" In *proc. of COLING'96*, pp.412–417. 1996.

[9] T.Shimizu, H.Yamamoto, H.Masataki, S.Matsunaga, Y.Sagisaka: "Spontaneous Dialogue Speech Recognition using Cross-word Context Constrained Word Graphs", ICASSP96,pp.145–148 1996.

[10] T.Morimoto et al.: "A Speech and language database for speech translation research" In *Proc. of ICSLP'94*, pp.1791–1794, 1994.

第 6 章

著者の関連発表資料

○ 脇田由実、河井淳、飯田仁” 意味的類似性を用いた不適格な音声認識候補の検出について” 人工知能学会全国大会 (第 10 回) 1996-6

○ 脇田由実、河井淳、飯田仁” 意味的類似性を用いた音声認識正解部分の特定法と音声翻訳手法への応用” 人工知能学会、言語、音声理解と対話処理研究会 (第 17 回), pp.7-12, 1997-1

○ 脇田由実、河井淳、飯田仁” Correct parts extraction from speech recognition results using semantic distance calculation, and its application to speech translation” ACL/EACL Work-Shop Spoken Language Translation, pp.24-31, 1997-7

○ 脇田由実、河井淳、飯田仁” 意味的類似性を用いた後処理的な音声認識正解部分特定法と音声翻訳手法への導入” 音声言語情報処理研究会 (第 17 回), 1997-7

第 7 章

参考資料 (ツール関連)

本研究開発に関連したツールの詳細を述べる。デモシステムに関する部分については、東洋情報システム（株）の河井氏の報告書を引用した。翻訳結果評価ツールについては、評価を担当して頂いた第三研究室の中祢氏、現在マンパワー（株）の栗田氏の報告資料を引用した。

7.1 正解部分特定法を導入したデモ用日英、英日音声翻訳システム

7.1.1 97 年度デモシステム概要

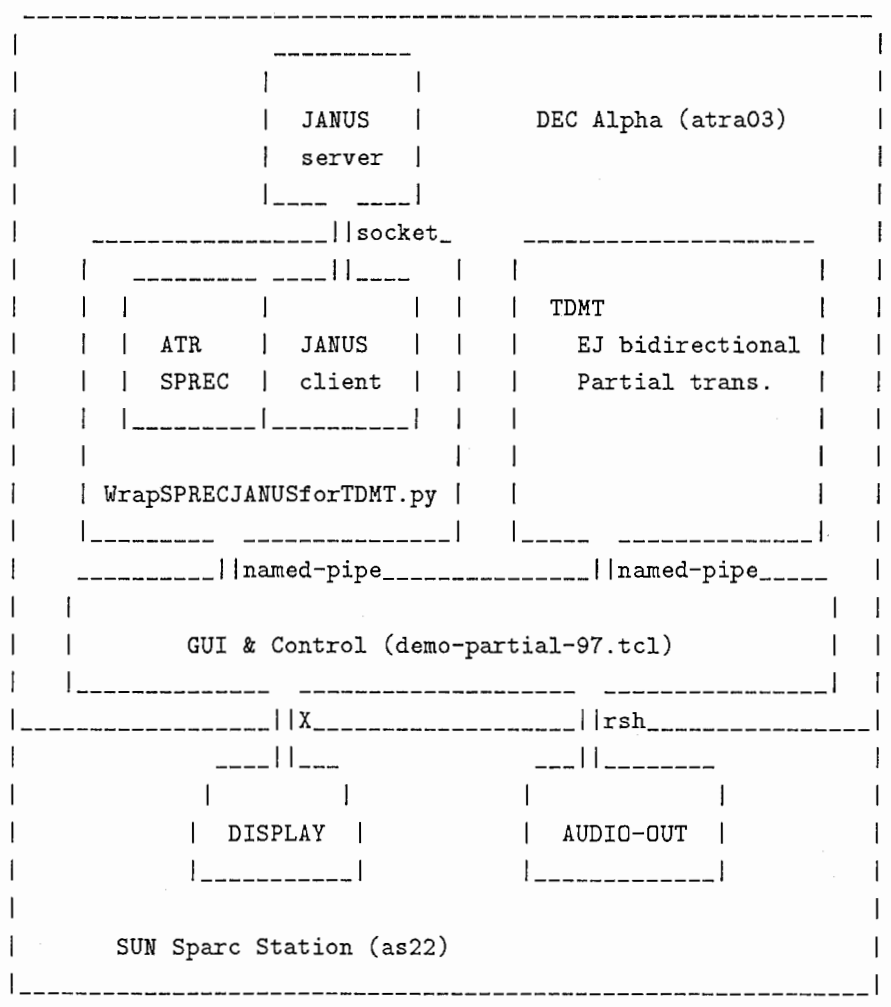


図 7.1 デモ用音声翻訳システム

特記事項

・音声認識モジュール (WrapSPRECJANUSforTDMT) との接続は
first-best 形態素列による。

・形態素の体系は EJ ともに TDMT 体系を使用。

・音声合成モジュール (CHATR) との接続は今回は見送った。

・処理本体は Alpha、 DISPLAY と AUDIO-OUT は SUN を用いた。

本デモシステムでは、音声認識結果を翻訳処理に渡し、音声認識正解部分の特定処理によって部分的に翻訳される様子を見る（あるいは示す）ことができる。音声認識部は第一研究室で開発されている ATR-SPREC、言語翻訳部は第三研究室で開発されている TDMT を使用した。検討課題である「理解可能な部分文の翻訳処理」は TDMT の一部を変更する形で実装されている。なお、音声認識の結果、および翻訳処理の結果は音声合成処理を通じて実際に音声として確認することも検討したが、時間の都合と技術的な課題のため、今回はこの機能を省くこととなった。

図 7.1 は 1997 年 11 月 6 日、7 日のオープンハウスでのデモシステム「理解可能な部分文のみ翻訳する日英・英日音声翻訳手法」の構成を示す。

本デモシステムとして提供される範囲は以下のとおりである。

- 部分翻訳のデモ画面
- 英日双方向の対応
- TDMT の変更部分（理解可能な部分文の翻訳処理）
- ATRSPREC, TDMT とのインターフェイス

なお、本デモシステムが動作するためには少なくとも以下の動作環境が必要である。

- csh, perl(ver4.0), gawk(ver2.15), tcl/tk(ver4.0), X window

さらに、本デモシステムが動作するためには以下のものを別途用意する必要がある。

- ATRSPREC, JANUS, python
- TDMT(tdmt-multi-970207 版), allegro-common-lisp(ver4.3)

上記のものが用意できないか、もしくは障害があり使用できない場合にはダミーファイルを用いて動作させることも可能である。但し、この場合にはダミーファイルに登録された順でしかデモできない。

7.1.2 インストールと設定

本デモシステムでは、X window による GUI でユーザインターフェイスを構成し、主にマウスによって操作するように設計されている。

インストール

プログラム及び、データは 8mm テープから UNIX の tar 形式のアーカイブにより供給される。次のシェルコマンドにより、テープから特定のディレクトリにインストールできる。（c-shell により demos ディレクトリにインストールする場合の例）

```
user@host[111]: mkdir  demos
user@host[112]: cd  demos
user@host[113]: tar  xvf  /dev/rst0  ./tdmt970207
```

設定

前述のとおり、別途用意した各プログラムと連携させなければならないため、これらを指定する必要がある。また、ユーザのマシン環境に合わせてカスタマイズすることもできる。変更可能な項目は、tdmt970207/demo-partial-97/config.txt のファイルに記述されているので、エディタなどで変更することができる。以下にその項目を解説する。

- 全般の設定 Top_Dir： デモシステム本体のインストール先のディレクトリ
  Trans_Com_List_File： 翻訳プロセスへのコマンドファイル
  Wave_List_J_File： 日本語の入力波形ファイルを指定するリストのファイル
  Wave_List_E_File： 英語の入力波形ファイルを指定するリストのファイル
  Label_List_File： デモ画面のラベルを表わすリストのファイル
  Label_List_E_File： 英語版のラベルを表わすリストのファイル
  Label_List_J_File： 日本語版のラベルを表わすリストのファイル
  Color_List_File： 表示色を指定するリストのファイル

  Default_Font： ディフォルトのフォント指定
  Default_KanjiFont： ディフォルトのフォント指定（日本語）
  Text_Font： テキストのフォント指定
  Text_KanjiFont： テキストのフォント指定（日本語）

  Frame_Size： デモ画面のサイズ（LARGE: 大きい、SMALL: 小さい）

- 音声入力プロセスの設定 Wave_Run_Mode： 実プロセスを実行するかどうか（REAL: する、DUMMY: しない）
  Wave_In_REAL： 実プロセスを実行する場合のプロセス起動
  Wave_In_DUMMY： 実プロセスを実行しない場合の処理（特になければ、とする。）

- 音声認識プロセスの設定 Sprec_Run_Mode： 実プロセスを実行するかどうか（REAL: する、DUMMY: しない）
  Sprec_In_REAL： 実プロセスを実行する場合のプロセス起動（入力側）
  Sprec_Out_REAL： 実プロセスを実行する場合のプロセス起動（出力側） 通常入力側と同じプロセスの場合はとする。
  Sprec_Server_REAL： 実プロセスを実行する場合のサーバープロセス起動（JANUS）
  Sprec_In_DUMMY： 実プロセスの代わりの代替プロセス（通常入力ファイルオープン）
  Sprec_Out_DUMMY： 実プロセスの代わりの代替プロセス（通常出力ファイルオープン）

- 翻訳プロセスの設定 Trans_Run_Mode： 実プロセスを実行するかどうか（REAL: する、DUMMY: しない）
  Trans_In_REAL： 実プロセスを実行する場合のプロセス起動（入力側）
  Trans_Out_REAL： 実プロセスを実行する場合のプロセス起動（出力側）
  Trans_Process_REAL： 実プロセスを実行する場合のプロセス起動（翻訳サーバー起動）
  Trans_In_DUMMY： 実プロセスの代わりの代替プロセス（入力側）
  Trans_Out_DUMMY： 実プロセスの代わりの代替プロセス（出力側）
  Trans_Process_DUMMY： 実プロセスの代わりの代替プロセス（翻訳サーバー起動）

- 音声合成プロセス Spsyn_Run_Mode： 実プロセスを実行するかどうか（REAL: する、DUMMY: しない）

Spsyn_In_REAL：　実プロセスを実行する場合のプロセス起動（入力側）
Roman_Flt_J_REAL：日本語からローマ字への変換フィルタ
Spsyn_In_DUMMY：　実プロセスの代わりの代替プロセス（入力側）
Roman_Flt_J_DUMMY：日本語からローマ字への変換フィルタの代替プロセス

7.1.3　操作説明

起動および終了方法

実行マシンへログインし、環境変数にディスプレイを指定する。

```
user@host[130]: setenv DISPLAY as06:0.0
```

プログラムは次のシェルコマンドで起動できる。内部的にいくつかのプロセスを起動するため、完全に起動するには少し時間がかかる。プロセスが完全に起動し終わっていなくてもマウスからボタンを押すなどしてコマンドを発行することはできる。この場合、起動され次第、コマンドが実行される。コマンドが実行される様子は、起動したシェルに出力される。

```
user@host[131]: cd  demos/tdmt970207
user@host[132]: csh  -f  demo-partial-97.csh
```

メニューの「ファイル」から「終了」を選択するか、同じシェルで Ctrl+C とすることによりデモシステムを終了することができる。通常、この操作ですべてのサブプロセスを終了できるが、ごくまれに終了できない場合がある。その場合は、サブプロセスを kill コマンドにより終了させる必要がある。主なサブプロセスの ID は次のようにして知ることができる。これらを kill すればその他のものは自然に消滅する。

```
user@host[134]: ps auxw | grep TDMT
user@host[135]: ps auxw | grep WrapSPRECJANUSforTDMT.py
```

画面操作の詳細

○ メニュー

- ファイル
  「終了」　　デモシステムを終了する。
- 選択
  入力（波形ファイル）を選択する。
- オプション
  現在、「音声入力」「音声認識処理」は機能無し。　「翻訳処理」　　現在「部分翻訳」のみサポート。　「部分翻訳処理」　　「列表示」「木構造表示」の切り替え。

○ ボタン

- 翻訳方向の切替え
  上部のラジオボタンにより、翻訳方向を切替えることができる。翻訳方向は、英日と日英のどちらかを選べる。
- 音声認識実行の切替え
  上部のラジオボタンにより、音声認識実行を切替えることができる。これは実際に音声認識処理を実行するのか、あるいはあらかじめ得られた音声認識結果を利用するのかを選択する。
- 音声入力
  メニューから「選択」された音声入力をスピーカで確認する。

・ 音声認識処理
音声入力の波形ファイルを音声認識処理に入力し、認識結果を表示する。

・ 部分翻訳なし
音声認識結果を翻訳処理に入力し、翻訳結果を表示する。

・ 部分翻訳あり
部分翻訳処理を実行した場合の結果を表示する。　（部分翻訳処理の実行は上記の翻訳処理の中で既に行われている。）

○ パラメータ設定

・ 最大距離
用例距離計算に対する制限を与えるための閾値。

・ 最小単語数
短い単語列に対して採否を決めるための閾値。　（詳細は「ガーベージ区間の判定」参照）

7.1.4 実装説明

本デモシステムは、本体がTcl/Tkにより書かれており、トップレベルはcshのスクリプトから呼ばれるようになっている。音声波形出力、音声認識、言語翻訳、音声合成の各処理はサブプロセスとして外部化されており、パイプによりデモシステム本体と接続される。この方式のため、適当なダミーファイルを用意してパイプに接続することにより、部分的なテストを容易に行うことができる。また、各プロセスはそれぞれC言語■Common Lisp言語を始めとして様々なスクリプト言語を利用して書かれているが、これに関係なく標準入出力を利用してデモシステム本体と接続している。デモシステム本体では、基本的にデータの流れを制御し、表示に必要な情報を得ているだけであり、各プロセス間に渡される情報の詳細については関知しない。

プログラムファイル

インストール先のディレクトリにあるdemo-partial-97.cshがトップレベルのシェルスクリプトである。ここから画面周りのTcl/Tkスクリプトであるdemo-partial-97/demo-partial-97.tclを呼び出すことによって各種サーバーが呼び出され、プログラムが起動する。

設定ファイル

config.txt： マスター設定ファイル

color_list.txt： テキスト色設定ファイル

label_list.txt： 画面上のラベルの指定ファイル

label_list_j.txt： 画面上の日本語ラベルの指定ファイル

label_list_e.txt： 画面上の英語ラベルの指定ファイル

trans_com_list.txt： 翻訳処理へのコマンドの指定ファイル

wave_list_j.txt： 日本語の入力波形の指定ファイル

wave_list_e.txt： 英語の入力波形の指定ファイル

サブプロセスの呼び出し

- 音声入力プロセス
  音声波形ファイルを入力としてスピーカから音声を出力する。ディフォルトでは../bin/WAVE/Talk3を使用している。このプロセスは毎回起動され、処理が終われば終了する。

- 音声認識処理プロセス
  音声波形ファイルを入力として音声認識プログラムを実行する。ディフォルトでは
  bin/SPREC/WrapSPRECJANUSforTDMT.py
  が音声認識サーバーとして使用される。これは第一研究室で開発されたATRSPRECおよび、MITで開発されたJANUSを使用して、日英双方の音声認識を実現している。音声波形ファイル名を標準入力で受け取り、音声認識結果を標準出力する。(今回はn-best出力の第一候補を利用)デモシステムの本体であるdemo-partial-97.tclは、read/writeモードでこのプロセスと接続して入出力を行う。なお、英語、日本語の切替えは、波形ファイル名の代わりにlanguage=ENGLISHまたは、language=JAPANESEの行を標準入力で受けることにより動的に行なわれる。なお、JANUSはそれ自体がさらにサーバークライアントとなっているので、
  bin/SPREC/WrapSPRECJANUSforTDMT.py
  はSPREC本体とJANUSクライアントを統合したものである。このため、Sprec_Server_REALによってJANUSサーバーを別に起動している。

- 翻訳処理プロセス
  音声認識処理の結果を入力として言語翻訳プログラムを実行する。ディフォルトではaf.cshを入力用プロセスとし、tdmt_of.cshを出力用プロセスとしている。翻訳処理はこれらの入出力プロセスと名前付きパイプ(named pipe)で接続されているTDMT-MULTI.cshが行う。これらのサブプロセス群は初めに一度起動されると、受けたデータを順次処理するようになっている。デモシステム本体は入出力プロセスを通じてデータをやりとりする。

- 音声合成処理プロセス
  前述のとおり、今回はこの機能を使用していない。

音声合成処理について

当初、音声合成(CHATR)の使用を検討したが、主に次のような理由により、今回はこの機能を省くこととなった。

(1) 前回と異り、DEC Alpha上でサーバーを動かす必要があった。このとき、CHATRは(a)Alpha、または(b)Sun上で動かす方法が考えられる。しかし、これらはどちらも次の課題を生じる。

  (a) Alpha上の場合：CHATRの音声出力をSunに送る必要がある。

  (b) Sun上の場合：CHATRへの入力をSunに送る必要がある。このため、データの受渡しなどに関する技術的課題が生じる。

(2) CHATRの最新版への対応が間に合わない。データの受渡しの仕様を検討する時間が十分に取れない。

7.2 翻訳結果評価ツールと作業手順

音声認識出力文を翻訳した場合の翻訳評価ツールと評価手順について述べる。評価手順の概要を図7.2, 図7.3に示す。

7.2.1 評価作業手順

（1） `ORDER=1` を取り出す

図7.2

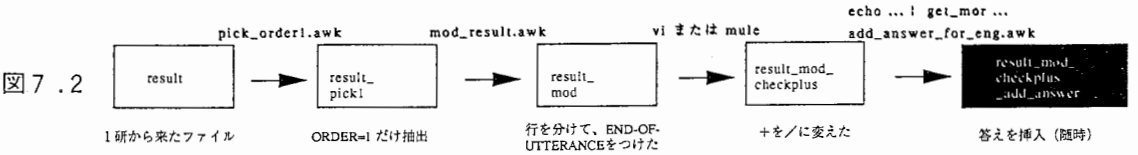
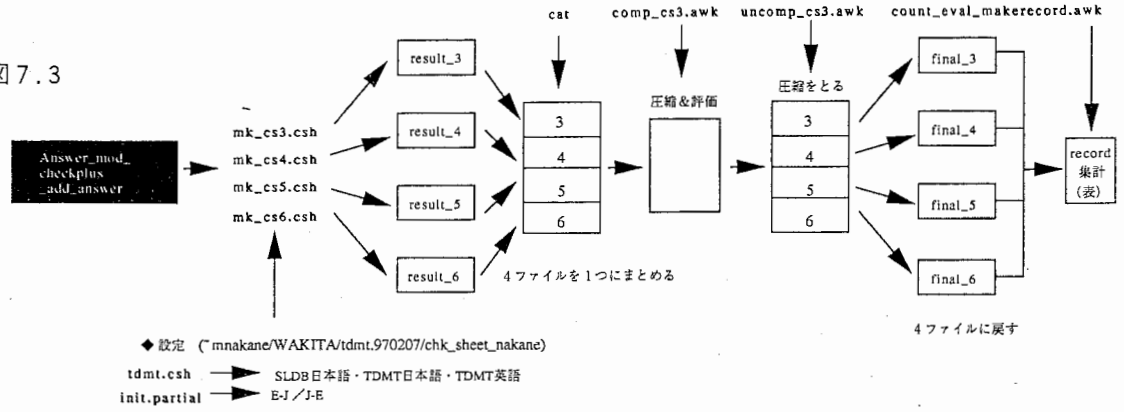


図7.3



第1候補文(ORDER=1)を翻訳する際には、まず1研から渡されたファイルを見て、ORDER=1, ORDER=2 など複数の認識結果が含まれているようなら、 ORDER=1 だけを取り出したファイルを作る作業を最初にしなければならない。

使用コマンド： ~mnakane/CMND/pick_order1.awk [A] > [B]
A というファイルの各 UTTERANCE から ORDER=1 だけを取り出して、
B というファイルにする。

(2) 行を分けて、END-OF-UTTERANCE の行を加える
使用コマンド：mod_result.awk [A] > [B]
[A] というファイルの行を項目ごとに分れさせて、最後に comment=END-OF-UTTERANCE- の行を付け加え、[B] という名前にする。
所在：~mnakane/CMND/
~mnakane/CMND/mod_result.awk wd.adapt.15.90.result > wd.adapt.15.90.result_mod

(3) 『+』 を 『/』に変える
上記ファイル中にある『+』を『/』に変え、そのファイル名を wd.adapt.15.90.result_mod_checkplus とする。
% cp wd.adapt.15.90.result_mod wd.adapt.15.90.result_mod_checkplus

(4) 正解音声ファイルの作成と挿入

(4-1) 正解音声の作成
使用コマンド：echo [A] |
get_mor -X1 -X2 -X3 -LEX [B] -mor_path [C] -sn > [D]
それぞれの項目で指定するのは、

A ：会話 ID (例：TAS12008)

-X1 ：A から取りだしてきたい言語 （＝ここでは source 言語）
(-j or -e つまり日英なら j、英日なら e)
-X2 ：使用する言語体系（-dept3 or -dept4 つまり TDMT か SLDB か）
-X3 ：どちらの役をとりだすか
(-a or -b or 指定なし つまり客かホテル側か両方か)
B ：-X1 で指定した source 言語の -X2 で指定した体系の辞書ファイル
日本語 SLDB 体系：/dept1/work7/MASTER_LEX/MASTER_LEXICON
日本語 TDMT 体系：/dept3/work22/yumi/TANIGAKI/LM4/MASTER_LEXICON_DEPT3
英語 SLDB 体系 ：今のところなし \\
英語 TDMT 体系 ：/dept3/work22/yumi/Megumi_work/MLEX_E/MLEX_E_970808.txt
C ：-X1 で指定した source 言語の -X2 で指定した体系の形態素ファイルの入っているディレクトリ
日本語 SLDB 体系：/dept1/work7/MASTER_LEX/JMOR/SLDB_LNG/
日本語 TDMT 体系：/dept3/work22/yumi/TDMT_JMOR_SLDB_JOE_970604/
英語 SLDB 体系 ：今のところなし
英語 TDMT 体系 ：/dept3/work22/yumi/EMORT_970808/
D ：出力ファイル名

これで、正解音声が数字に置き換えられて、正解音声ファイルが作成される。
＊注意：今のところ、英語 SLDB 体系での認識結果については、このコマンドでは正解音声をつくることができない。

(4-2) 正解音声の挿入
使用コマンド：add_answer_for_eng.awk [A] [B] > [C]
A というファイルにまとめた word id no （正解音声）を B というファイルに挿入して、C という名前で保存する
所在：~mnakane/CMND\\
＊名前は for_eng だが、E→J だけでなく J→E にも使える。

7.2.2 評価ツールの起動 ー圧縮から結果を出すまでー

（1） 前準備：設定の確認
コマンドを起動させる前に設定を確認する。
（設定内容は毎回異なるので随時確認する）

```
% cd ~mnakane/WAKITA/tdmt.970207/chk_sheet_nakane/

init_partial.tdmt の設定
                                                        J-E     E-J
TDMTCommand="0 (switch-mode :J-E)"   ←主な変更箇所。今回は日→英 英→日は逆
TDMTCommand="0 (set-nbest-number 1)"
TDMTCommand="0 (switch-translation-method :partial)"
TDMTCommand="0 (switch-output-function 'output-partial-result)"
TDMTCommand="0 (set-output-tree nil)"
TDMTCommand="1 (setq *print-pretty* nil)"
TDMTCommand="21 (set-partial-max-dist .2)"
TDMTCommand="21 (set-partial-min-len 2)"
```

tdmt.csh の設定

```
#!/bin/csh -f
#setenv ALLEGRO_CL_HOME /usr8/Allegro_Common_Lisp4.3/home;
#set A_filter = af.csh     # SLDB 体系日本語形態素
set A_filter = af_3.csh  # TDMT 体系日本語形態素
#set A_filter = af_e.csh  # TDMT 体系英語形態素
```

（2）翻訳結果を出す

翻訳結果を出すコマンド：mk_csX.csh

```
% cd ~mnakane/WAKITA/tdmt.970207/chk_sheet_nakane/
mk_cs3.csh 認識結果ファイル > 翻訳結果ファイル1
mk_cs4.csh 認識結果ファイル > 翻訳結果ファイル2
mk_cs5.csh 認識結果ファイル > 翻訳結果ファイル3
mk_cs6.csh 認識結果ファイル > 翻訳結果ファイル4
```

このコマンドで、(I) の作業で作られたファイルの分割、部分翻訳結果を出す。
各コマンドの設定の違いは、添付資料1を参照。

```
% mk_cs3.csh ~mnakane/WAKITA/Eval_TDMT/wd.adapt.15.90.result_mod_checkplus_add_answer >
  ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_3_970605_checkplus

% mk_cs4.csh ~mnakane/WAKITA/Eval_TDMT/wd.adapt.15.90.result_mod_checkplus_add_answer >
  ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_4_970605_checkplus

% mk_cs5.csh ~mnakane/WAKITA/Eval_TDMT/wd.adapt.15.90.result_mod_checkplus_add_answer >
  ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_5_970605_checkplus

% mk_cs6.csh ~mnakane/WAKITA/Eval_TDMT/wd.adapt.15.90.result_mod_checkplus_add_answer >
  ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_6_970605_checkplus

% cd ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised

% ls
result_3_970605_checkplus      result_5_970605_checkplus
result_4_970605_checkplus      result_6_970605_checkplus
```

（3）翻訳結果をひとつにまとめる

cat コマンドで、出来上がった4ファイルを1ファイルに
(result_all_970605_checkplus) まとめる。

```
% cat ファイル名 > まとめたファイル

% cat result_?_970605_checkplus > result_all_970605_checkplus

% ls
chk_readme_970605              result_5_970605_checkplus
result_3_970605_checkplus      result_6_970605_checkplus
```

```
result_4_970605_checkplus        result_all_970605_checkplus

% grep 'END-OF-UTTERANCE' result_all_970605_checkplus | wc
     252     252    7108
```

データが全部で何項目あるか調べる。この場合、63X4=252 ある。

（4）圧縮

使用コマンド：comp_cs3.awk*

```
% cd ~mnakane/WAKITA/Eval_TDMT/JAPANESE
comp_cs3.awk まとめたファイル > 圧縮ファイル
```

これで result_all_970605_checkplus を圧縮して result_all_comp_970605_checkplus とする。

```
% comp_cs3.awk 1VER_BUNKATU_BUBUN_Revised/result_all_970605_checkplus >
  1VER_BUNKATU_BUBUN_Revised/result_all_comp_970605_checkplus

% cd ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/

% ls
chk_readme_970605                result_6_970605_checkplus
result_3_970605_checkplus        result_all_970605_checkplus
result_4_970605_checkplus        result_all_comp_970605_checkplus
result_5_970605_checkplus

% grep 'END-OF-UTTERANCE' result_all_comp_970605_checkplus | wc
     133     133    3755
```

圧縮することで 252 項目が 133 項目になる。

（5）評価

できあがったファイル result_all_comp_970605_checkplus の翻訳結果を評価し、入力する。

（6）圧縮解除

使用コマンド：uncomp_cs3.awk

```
% cd ~mnakane/WAKITA/tdmt.970207/chk_sheet_nakane
uncomp_cs3.awk まとめたファイル　圧縮ファイル　> 新しいファイル名
```

これで、圧縮したファイルをもとに戻して、result_all_uncomp_970407 とする。

```
% uncomp_cs3.awk ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_all_97040
  ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_all_comp_970407 >
   ~mnakane/WAKITA/Eval_TDMT/JAPANESE/1VER_BUNKATU_BUBUN_Revised/result_all_uncomp_970407
```

さらにその result_all_uncomp_970407 を mule 上で 4 ファイルに手作業で戻す。

```
% ls
result_3_final_970605_checkplus
result_4_final_970605_checkplus
result_5_final_970605_checkplus
result_6_final_970605_checkplus
```

（7）結果集計

使用コマンド：count_eval_makerecord.awk

```
% cd ~mnakane/CMND/
count_eval_makerecord.awk 集計したいファイル名 > 集計結果ファイル
```

これで集計結果が次のように出てくるので、4ファイルすべての結果を出す。。

```
 # (1) : 23 ( 0.365079 ) (%)
 # (2) : 0 ( 0.000000 )
 # (3) : 4 ( 0.063492 )
 # (4) : 8 ( 0.126984 )
 # nil : 23 ( 0.365079 )
 # NIL : 5 ( 0.079365 )
```

```
% count_eval_makerecord.awk JAPANESE/1VER_BUNKATU_BUBUN_Revised/MEGUMI/
  result_3_final_970605_checkplus
  > JAPANESE/1VER_BUNKATU_BUBUN_Revised/MEGUMI/record
```